XenDesktop 7.1

Mar 07, 2016

About XenDesktop

Known issues
Components for this release
Features not in this release

XenDesktop administrators

System requirements

Platform support

How do I...

Get started

XenDesktop concepts
Important information for XenApp administrators

Plan

Components
Example deployments
Database fault tolerance
Desktop and application access if Delivery Controllers fail
Security planning
Host desktops and applications
Evaluate applications
Enhance the user experience for mobile devices
Active Directory

License

System requirements for Citrix Licensing
Manage licensing

Install

Install using the graphical interface
Install using the command line
Install VDAs using scripts in Active Directory
Create a Site
Install a VDA on earlier systems
Remove components

Upgrade XenDesktop 5
XenDesktop 5 upgrade components
XenDesktop 5 upgrade factors

Upgrade XenDesktop 7

Migrate
Data import and export details
Export from a XenDesktop 4 farm
Edit the Migration Tool XML file
Import XenDesktop 4 data
Migration examples
Post-migration tasks
XenDesktop 4 components that are not migrated

Deliver
Desktop and application delivery
Set up machine catalogs
Delivery Groups
Hosted applications
High definition user experience with HDX
Enhance the user experience for mobile devices
Mobility features
Provide users with Remote PC Access
Prevent user access to the local desktop
Server VDI

Manage
Machines
Application and desktop delivery
Delivery Controller environment
Change secondary database locations
Connections and resources
Manage Citrix policies
User profiles
Maintain session activity
Personal vDisks
Configuration Logging
Delegated Administration
IPv6
Configure time zone settings
Configure connection timers
Configure USB support
Support for USB mass storage devices
Client folder redirection

Print
Plan your printing configuration
Provision printers
Manage printer drivers
Optimize printing performance
Display printers and manage print queues

Monitor
Monitor environments with Director
Monitor Personal vDisks

Integrate
Microsoft System Center Virtual Machine Manager and virtual machine management
Properties for Microsoft System Center Configuration Manager
VMware and virtual machine management

Secure
Authenticate securely with smart cards
Plan pass-through authentication with smart cards

Reference
About the XenDesktop SDK
Policy settings reference
Use the Monitor Service OData API

Third party notices
About XenDesktop

Mar 28, 2015

This document provides information about the new components and features supported in this release.

Review the System Requirements and list of Known Issues in this document carefully before you begin your planning.

These new Citrix and Microsoft releases can be used with this release:

- **Citrix AppDNA** (see the AppDNA documentation). Accelerates the migration and transformation of desktop and web applications for new environments and helps manage application change on a day-to-day basis. AppDNA provides automated analysis of applications for different Windows platforms and suitability for application virtualization through App-V, XenApp, or XenDesktop. Using AppDNA, you can model complex mixes of new technologies to determine the best plan of action and then automate the transformations.

- **Citrix StoreFront** (see the Technologies > StoreFront documentation). This feature replaces Web Interface and provides authentication and resource delivery services for Citrix Receiver.

- **Citrix Receiver** (see the Receiver documentation for user devices and platforms). Receiver is a universal software client that provides secure, high-performance delivery of virtual desktops and applications.

- **Citrix Personal vDisk improvements** (see Personal vDisks in the XenDesktop documentation). This feature retains the single image management of pooled and streamed desktops while allowing people to install applications and change their desktop settings.

- **Citrix Provisioning Services** (see the Technologies > Provisioning Services documentation). This feature allows computers to be provisioned and re-provisioned in real-time from a single shared-disk image.

- **Microsoft System Center Configuration Manager (SCCM) 2012 R2**. This Microsoft application deployment tool provides the same functionality delivered by SCCM 2012 SP1. However this release does not support new SCCM 2012 R2 features.

- **Microsoft System Center Virtual Machine Manager (VMM) 2012 R2**. These hypervisor versions are supported by Provisioning Services and provide improved performance and scalability with Hyper-V. This release also provides performance improvements for Machine Creation Services (MCS) when using Server Message Block (SMB) 3.0 on file servers with clustered shared volumes and Storage Area Networks (SANs).

For information about supported languages, see CTX119253

What’s new in XenDesktop 7.1

- **GPU** — The Graphical Processing Unit (GPU) capabilities feature provides graphics virtualization, offering a superior experience for users who use graphic-intensive applications and often manipulate 3-D models. The GPU feature also can improve the graphics experience for a large number of users. Administrators can evaluate workflows for the creation of GPU-capable connections and machine catalogs created with an MCS-provisioned master image. If problems are encountered, then Administrators may need to manually create the machines.

- **vGPU** — The Virtual Graphical Processing Unit (vGPU) feature enables multiple virtual machines to directly access the graphics processing power of a single physical GPU. You can use hardware-accelerated vGPU access for Windows desktop VDI workloads. This true hardware GPU sharing provides full Windows 7 or Windows 2008 R2 SP1 desktops suitable for users with complex and demanding design requirements. Supported for NVIDIA GRID K1 and K2 cards, the GPU sharing uses the same NVIDIA graphics drivers that are deployed on non-virtualized operating systems.

- **Windows Server 2012 R2 and Windows 8.1 support** — Deliver a high-definition user experience on Windows 8.1 virtual desktops and physical machines. HDX 3D Pro has also been upgraded to support Windows 8.1.
Note: The vGPU feature is not supported for virtual machines running Windows Server 2012 R2 or Windows 8.1.

What's new in XenDesktop 7

- **Machine Catalog Desktops and Applications**
  - **Desktops** — Deliver managed desktops to multiple, simultaneously connected remote users. Server OS machine and Desktop OS machine desktops provide features such as SmoothRoaming, session reliability, license consumption, Receiver and Delivery Services authentication, and Desktop Viewer. This release introduces Windows Server OS machine catalogs and desktops as a cost-effective virtualization delivery solution because they provide a significant number of remote users per server and storage resource. This release also provides Remote PC desktops that let users access resources on their office PCs remotely, from any device running Citrix Receiver.
  - **Applications** — Integrate XenApp publishing capabilities within a deployment. Use this feature to deliver shared hosted applications to multiple, simultaneously-connected remote users.

- **HDX enhancements**
  - **Windows Server 2012 and Windows 8 support** — Deliver a high-definition user experience on Windows 8 virtual desktops and physical machines. HDX 3D Pro has also been upgraded to support Windows 8.
  - **Desktop Composition Redirection** — Extend the Aero Redirection feature introduced in XenDesktop 5.5 to provide users with a more fluid Windows 7 Aero or Windows 8 desktop experience by leveraging the graphics processing unit (GPU) or integrated graphics processor (IGP) on supported Windows user devices for local DirectX graphics rendering. This feature delivers rich multimedia applications to user devices, while maintaining high scalability on the server.
  - **Windows Media client-side content Fetching** — Enable a user device to stream multimedia files directly from the source provider on the Internet or Intranet to improve network utilization and server scalability.
  - **Multicast support** — Reduce bandwidth consumption when streaming live video to branch offices. Multicast support enables a single Windows Media source transmission to support multiple users.
  - **Real-time multimedia transcoding** — Enable audio and video streaming to mobile devices, improving the way that Windows Media content is delivered. Host-based transcoding provides a seamless user experience, even in extreme network conditions. To improve server scalability, if the Virtual Delivery Agent (VDA) has a supported graphics processing unit (GPU) for hardware acceleration, transcoding is done in the GPU.
  - **User Datagram Protocol (UDP) audio for Server OS machines** — Extend support for audio delivery over UDP/RTP to Server OS machines. This feature delivers superior audio quality for real-time applications like video conferencing and streaming media, even in environments when there is packet loss or congestion.
  - **Webcam video compression** — Reduce bandwidth consumption to improve performance when using supported video conferencing applications.
  - **HDX 3D Pro** — Deliver applications with graphics processing units (GPUs) for hardware acceleration to the desktop. This includes 3D professional graphics applications based on OpenGL and DirectX.
  - **Server-rendered Rich Graphics and Video** — Deliver rich graphics (including Windows 8 or Windows Aero) to virtual desktop users on any client, platform, and bandwidth condition by rendering the content on the server. When users have the latest Citrix Receiver, they will also notice improved performance for server-rendered video.
  - **Improved Flash Redirection** — Determine when to redirect Adobe Flash content to the user device for local rendering based on the user's network type and environment. Flash Redirection helps reduce server and network load, resulting in greater scalability while ensuring a high-definition user experience.
  - **New installer** — Use a single, streamlined installer to guide you through installing the core components (Delivery Controller, Studio, Director, StoreFront, and License Server) and VDAs.
  - **Profile management** — By default, Citrix Profile management 5.0 is installed silently on master images when you install the Virtual Delivery Agent, but you do not have to use Profile management as a profile solution.
  - **Configuration Logging** — Capture Site configuration changes and administrative activities to a Configuration Logging Database. You can view the log in Studio using a variety of filters and generate HTML and CSV reports.
  - **Director** — Monitor and troubleshoot deployments using Director's redesigned user interface with integrated EdgeSight
features:

- A new help desk view offers an improved troubleshooting experience for help desk administrators, allowing user, machine, and application issues to be resolved quickly.
- Full administrators have access to other views, including a newly designed Dashboard to provide a graphical summary of your deployment in a central location, and a Trends page to provide improved, in-depth, graphical monitoring and troubleshooting of the entire deployment with various time ranges.
- EdgeSight performance management provides the historical retention and trend reporting. With historical retention of data versus the real-time assessment, administrators are able to create Trend reports, including capacity and health trending.
- EdgeSight network analysis leverages HDX Insight to provide an application and desktop contextual view of the network. With this feature, Director provides advanced analytics of ICA traffic in their XenDesktop deployment.
- Delegated Administration — Group objects into administrative scopes. This feature provides an enterprise-class administration model with role-based access control, custom roles with configurable permissions, and fine-grained, object-based control.
- StoreFront replaces Web Interface — Create centralized enterprise stores to deliver desktops, applications, and other resources to users on any device. This feature provides authentication and resource delivery services for Citrix Receiver.
- Delivery Controller auto-update — Automatically notify Virtual Delivery Agents (VDAs) when Controllers are added to and removed from the Site. This feature helps prevent VDA rejection of sessions that are launched by unknown Controllers, and VDA startup delays or errors caused by invalid Controller information. For information about how to preserve the CNAME functionality, see CTX137960.
- Client Folder Redirection — Change the way client-side files are accessible on the host-side session. When you enable only client drive mapping on the server, client-side full volumes are automatically mapped to host drive letters. When you enable client folder redirection on the server and then the user configures it on the user device, only the portion of the local volume that is specified by the user is redirected.
- Improved Virtual Desktop Access Control Settings – Control user access to both Server OS Machines and Desktop OS Machines in a simple and unified way with a new, streamlined group of security settings.
- Improved and integrated error reporting — Studio error reporting links directly to the Citrix Support website. When users encounter an error situation, choosing the Get Advice option submits information about the error to the Citrix website. The information is analyzed and the user is redirected to a Website containing remedial advice.
- StoreFront enhancements
  - Desktop Appliance sites — Access Desktop Appliance sites through a website on the StoreFront store. The site is created automatically when a store is created. If, for example, the store has a path of path/Citrix/Store, the Desktop Appliance site path is path/Citrix/StoreDesktopAppliance. Users can restart their virtual machine-hosted desktops on Desktop Appliance sites.
  - Database as a service — Write user subscription data for each store to the local disk on the StoreFront server by using the new subscription store service. The data is then propagated across the server group.
  - IPv6 support — Connect to clients and core components on IPv4, IPv6, or dual-stack (IPv4/IPv6) environments.
  - Personal vDisk — Personalize your virtual desktops. In addition, a dedicated storage disk is created, before logon, so that users can store their data on the desktop, including any applications they install.
  - Machine Creation Services (MCS) support for Microsoft Key Management System (KMS) activation — Each virtual machine (VM) created with MCS provides a unique activation for the Windows operating system and Office 2010, which enables the KMS system to record each VM as a separate machine.
  - Support for group policies configured in Citrix Mobility Pack — For details about these policies for Citrix Receiver for mobile devices, see Configure policies for mobility features.
  - Multi-touch support — Touch functionality is supported on multi-touch computers, including Microsoft Windows 7 and Windows 8 workstations and Windows Server 2012 for VDAs.
• **Remote PC Access** — Automated administration of Remote PC Access is fully integrated into the core functionality of the XenDesktop Delivery Controller and Studio. One Remote PC machine catalog and one Remote PC Delivery Group are created automatically when you set up a Remote PC deployment; you can add more machine catalogs or Delivery Groups later using Studio. This process replaces the XML configuration file and PowerShell scripts used by Remote PC Access in XenDesktop 5.6 FP1. The current release also adds the ability to remotely access office PCs running Windows 8.

• **Support for Fast User Switching using RDP connections** — This Microsoft Windows feature makes it possible for multiple users to share a desktop without closing programs or logging off existing users. Administrators can take advantage of this feature to troubleshoot problems and install updates, without interrupting tasks or programs in use by the logged on user, by initiating an RDP connection to the VDA.

• **Compatibility with AhnLab keyboard encryption used in Korea** — This release is compatible with keyboard encryption that is installed on Virtual Desktop Infrastructure (VDI) hosts.

• **Universal Print Server** — The Delivery Controller now includes the Universal Print Server functionality. You need only install the Universal Print Server on your print servers. The Universal Print Server includes support for Windows Server 2012 and Windows 8.

See also:

- Features not in this release
- Known issues
- Fixed Issues (Delivery Controller)
- Fixed Issues (Virtual Delivery Agent)
- Fixed Issues (Provisioning Services)
- Fixed Issues (Receiver for Windows)
- Fixed Issues (Profile management)
- StoreFront Known Issues and Fixed Issues
Known issues

Dec 12, 2014

General issues

The following note applies to any workaround that suggests changing a registry entry:

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

- In a multiple monitor environment with Windows Media Redirection either disable or not supported, video played on a non-primary monitor may result in an error. To avoid the error play the video on the primary monitor. [#523948]
- To configure a non-standard HTTP/SOAP port for the Universal Print Server web service, use PowerShell cmdlets to change the session printer policy. For information about configuring Group Policy settings, see the Group Policy SDK usage section in the About the XenDesktop SDK topic. To set the policy value, use:
  ```powershell
  Set-ItemProperty -Path 'LocalGpo:\Computer\Unfiltered\Settings\ICA\Printing\UniversalPrintServer\UpsHttpPort' -Name Value -Value <portnumber>. [#268593]
  ```
- After a Hyper-V host is un-paused, Microsoft System Center Virtual Machine Manager (VMM) might not update the overall host state immediately. This can affect the use of Machine Creation Services (MCS). If one Hyper-V host reports a paused state, that host will not be used to provision Virtual Machines (VMs); if all hosts report a paused state, catalog creation will fail. As a workaround, manually refresh the parent cluster node or Hyper-V host node. Also, running environment tests on the host will identify hosts reporting a paused state. [#285696]
- Brokering hosted applications on Desktop OS machines is not supported using the Remote Desktop Protocol (RDP). [#377108]
- Logging on to Windows 8 virtual desktops can be slow and a black screen might be presented to users during this process before the Start screen appears. This issue will likely be experienced with dedicated desktops when they are first assigned or with pooled-random desktops created with Provisioning Services. See http://support.citrix.com/article/CTX124877. [#364655]
- When deploying virtual desktop VMs using Provisioning Services, the XenDesktop setup wizard does not use more than one storage resource in the Hosting Infrastructure record, even when more storage resources are available. As a workaround, when using Provisioning Services for deployment, create Hosting Infrastructure records with only one storage record. [#400212]
- This release does not support mounting an .iso file when Client Drive Mapping (CDM) is configured for Windows 8 XenDesktop sessions. [#333111]
- If the Citrix Universal Print Server fails because of bad drivers, try running those drivers under printer driver isolation. For information about how to configure printer driver isolation, see MSDN: http://msdn.microsoft.com/en-us/library/windows/hardware/ff560836(v=vs.85).aspx. [#381460]
- The Enhanced Desktop Experience policy setting does not affect pre-existing user or administrator profiles. As a workaround, delete all pre-existing profiles before enabling/disabling the setting, and delete the profile used for VDA installation (after installing the VDA). If the built-in administrator's account was used for (Virtual Delivery Agent (VDA) installation, you cannot delete the profile. In that case, the user can choose the Citrix Enhanced Desktop Theme when logged on. [#363736]
- If monitor resolution is affected because the VDA desktop session resolution exceeds the client monitor resolution, change the resolution setting on the VDA desktop to the highest supported value for the attached monitor.
Alternatively, you can detach the monitor cables from the VDA graphics card. [#365877]

- Screen savers and the power-save option are disabled in XenDesktop sessions. Edit the registry and create the following DWORD value:

```
HKLM\Software\Citrix\Graphics\SetDisplayRequiredMode = 0
```

This change does not prevent the remote machine screen saver or power save mode from coming on. If the power save mode comes on, the remote session is not updated until the user provides input (mouse/keyboard), but the screen will not be blanked. [#380550]

-Attempting to use a PowerShell SDK New-ProvScheme command or any MCS command from a remote machine before setting the host admin address might result in an error. Set the admin address using the Set-HypAdminConnection command before running the New-ProvScheme command. [#336902]

-This release does not support using XenDesktop 7 with Microsoft RemoteFX vGPU feature in a Hyper-V host. Instead, use RDP to access RemoteFX functionality such as Hyper-V vGPU. [#375577]

-Director reports the number of GPU machines that have failed to start in the Machine Failure pane on the Dashboard page. However, this information is not displayed when you drill down to view details on the Filters page.

You can also view this information in the Historical trends graph. [#0434722]

- If more GPU machines are provisioned or assigned than the XenServer resource pool can support of the GPU type specified, GPU machines may fail to start. Users assigned to a machine (including those using Personal vDisk) that failed to start may not be able to access their virtual desktop through StoreFront. [#0434505]

- If you power on a GPU machine and it fails to start, you may see the following error message:

```
Exception 'Failure in PowerOn, PGPU_INSUFFIENT_CAPACITY_FOR_VGPU
```

This indicates that there are insufficient GPU Resources to start another machine that uses the GPU. [#434509]

- If multiple users log on to a Desktop OS machine using RDP and ICA protocols and a user locks his/her RDP session, users connecting through ICA cannot log on to the session. To prevent this issue, users should disconnect or log off RDP sessions when they are finished. [#392311]

- When you publish an application with a Windows 8 VDA host and the application shows non-ASCII characters in the notification area, other unrelated characters might also appear in the notification area after reconnection. To resolve this issue, log off from the session and then launch the application again. [#387963]

- The ability to disable session sharing through a registry entry is not supported in this release. As a result, session sharing is always enabled. If the registry key is set to disable session sharing, the first application launches, but subsequent applications do not launch. There is no workaround for this issue. [#383718]

- When a user reconnects to a disconnected session on Windows Server 2012, there is a memory leak in the Desktop Window Manager. If users connect and disconnect frequently to long-lived sessions, such as in the case of an employee using the same session from work and home, this can cause memory resources on the server to be reduced, eventually leading to slow response times and possibly even server failure. This is a third-party issue in the Microsoft code. For information, see: http://support.microsoft.com/kb/2855336 (to be published in July 2013). [#374261]

- When using Delegated Administration to copy an administrator, some permissions might not be copied. Check the role definition of the copy against the role definition of the original to ensure all permissions were replicated. Then, edit the role and add any missing permissions, if needed. [#396927]

- When joining or removing a Controller from a Site, all StoreFront servers in the cluster must be powered on. Otherwise, the operation fails for the StoreFront part, and Studio throws an exception that mistakenly suggests operation failure. In such cases, restarting Studio corrects the Site display. If a StoreFront server cannot be brought back online, use the StoreFront console from another machine in the cluster to remove StoreFront from the failed server. To rejoin the failed server to the cluster, either reinstall StoreFront or successfully run the following commands in sequence: C:\Program
In deployments where a SQL Server 2012 database is configured for high-availability using AlwaysOn and is used with XenDesktop, there may be an interval (while a database failover is in progress) when Site operations may fail; for example, VDA power management or session launches. If this occurs, retry the impacted operation after the failover is complete. [#394787]

Applications might fail to launch on user devices if you enable the profile streaming policy without setting antivirus exclusions for Microsoft Group Policy files, such as NTUser.pol. As a workaround, disable profile streaming or set antivirus exclusion policies for those users. For information, see http://support.microsoft.com/kb/822158. In particular, read the section "Turn off scanning of Group Policy related files" for details about excluding the NTUser.pol and Registry.pol files. [#399152]

When users on Server OS machines log off from a session, the load is not removed immediately from the load index. It might take up to five minutes, and if the machine is at full load, others cannot log on during that time. After about five minutes, the load returns to normal. [#397888, 397884]

Opening some data files by double-clicking them results in an empty file if the File Type Association (FTA) HandlerOpenArguments parameter uses a custom format. This happens when the FTA is created through the PowerShell instead of Studio. If this occurs, the user should open the file through the application commands. This associates the file with the application and enables the user to double-click the file to open it in the future. [#399276]

XenDesktop policy settings, based on the SmartAccess feature of NetScaler Gateway, may be incorrectly retained when a user subsequently reconnects locally. For example, a XenDesktop policy that prevents remote users from mapping or redirecting client drives may remain in force when that user reconnects through the LAN. [#392967]

The Help About topics fail when working with XenDesktop through Windows PowerShell 3.0. This is a third-party issue with Microsoft. Other help topics are unaffected. [#408866]

Windows Store Apps installed to the virtual desktop from the store will not launch after the virtual desktop restarts through a Restart or Shutdown-Start command. Any apps installed after the virtual desktop restarts also fail to launch. [#413003]

Printers connected to a user device might not appear in the Control Panel under Devices and Printers on a Windows 8 virtual desktop. This occurs because Windows 8 does not include a Citrix-suggested "HP Color LaserJet 2800 Series PS" PostScript printer driver. To work around this issue, install on the Windows 8 master image a suitable PostScript driver or the "HP Color Laserjet 2800 Series PS" printer driver from Windows Update. [#411397]

Start menu tiles might not appear when you log in to a Windows 8 machine if you have previously logged in and logged out of a Windows 8.1 machine of the same bitness. To prevent this issue, avoid using Windows 8.1 profiles on Windows 8 machines. [#422043]

For System Center Virtual Machine Manager (VMM) with a standalone host and SMB as the OS storage, creating a new catalog and adding machines to an upgraded catalog may fail. As a workaround, move the host into a cluster of one, create a new hosting unit, and try again. [#426143]

If the Profile management feature is enabled, logon scripts for sessions running Windows Server 2012 R2 or Windows 8.1 are delayed by five minutes by default. Once the session is available, the Logon duration for logon scripts step is not available in Director. The delay is controlled by the Configure Logon Script Delay policy (Enabled = 0). [#407978]

Desktops may not launch on computers running Windows 8.1 with Microsoft Software Update Management installed. To avoid this, in the Windows 8.1 Taskbar and Navigation properties dialog box, on the Navigation tab, select all the options in the Start screen section. [#404839]

If you receive the error "You cannot access this session because no licenses are available" when you try to connect through Remote Desktop Protocol to a brokered session on XenDesktop 7.1, disable these settings in C:\inetpub\wwwroot\Citrix\Store\App_Data\default.ica: [#422212 and #403855]

- RDPConnection=false
- RDP-RedirectDrives=false
• RDP-RedirectDynamicDrives=false
• XenDesktop cannot be configured if it is attached to a SQL Server instance with a case sensitive default collation. XenDesktop must be deployed with a SQL Server with a case-insensitive default collation. [#429302]

• When you use a XenServer host and attempt to power on a machine that has a GPU, XenDesktop Configuration Logging might erroneously indicate the operation completed successfully. To confirm the power on completion, select a Machine Catalog containing that machine and check the machine's state. [#434507]

• The value for the Persistent Cache Threshold policy setting is labelled incorrectly, as Kbps, in Studio. The correct value is bits per second (bps). [# 429478]

Installation issues

• Installing a Virtual Delivery Agent for Server OS might fail with error 1935 because of backward compatibility errors in the Microsoft Visual C++ 2005 Redistributable. See the Microsoft website or run Windows Update to check for fixes. [354833]

• After a successful VDA for Windows Server OS installation, but before the machine restarts, the Windows event log might contain several error messages (such as TermService 1035 or 1036, indicating the Terminal Server listener stack was down or the session creation failed). If there are no other installation failure indicators, you can safely ignore those event log messages. [#374134]

• When upgrading from XenDesktop 5.x, make sure that the XenDesktop 5.x Desktop Studio is closed before running the XenDesktop 7 upgrade. Otherwise, Desktop Studio may close unexpectedly during the upgrade. [#389374]

• When installing VDA version 5.6 on a 64-bit Windows Vista or Windows XP system, do not include the Citrix Receiver in that installation; otherwise, the installation might not complete. Follow this procedure:
  1. Install the VDA, but clear the Citrix Receiver checkbox in the installation wizard.
  2. Install Receiver (CitrixReceiverEnterprise.exe) from the Citrix Receiver and Plug-ins > Windows > Receiver folder on the XenDesktop 7 media.
  3. If you need the offline plug-in, install CitrixOfflinePlugin.exe from the Citrix Receiver and Plug-ins > Windows > Offline Plug-in folder on the XenDesktop 7 media. If you specify an installation location other than the default (C:\Program Files (x86)\Citrix), be sure the destination directory exists before starting the installation.

   If you mistakenly start a VDA installation with the Citrix Receiver checkbox enabled, use Windows Task Manager to end the running XenDesktop application. [#381625]

• When upgrading, an administrator who was disabled in XenDesktop 5.6 might move to XenDesktop 7 without a role or scope. Check the Administrators display in Studio and edit administrators, as needed. [#394765]

• After performing an upgrade on a Controller, close Studio and then restart it so that the system recognizes that the Controller has been upgraded. Register the upgraded Controllers by selecting the controller, and then clicking Register Controller. [#382898]

• If you enable the Profile management feature and users find that their default Windows 8 applications (such as Weather, News, and Bing) start the first time they log on, but not after subsequent logons, you might have to reconfigure this feature. This issue has been observed in environments where the user profile is not persisted and if the folder AppData\Local has not been excluded (the default). As a workaround, add the folders AppData\Local\Packages and AppData\Local\Microsoft\Application Shortcuts as exclusions. [#394802]

• During VDA installation or upgrade on Windows 7, you might see a Windows dialog box prompting you to Restart the computer to apply changes. Click Restart Later to continue the upgrade; do not select Restart Now. [#396553]

• Installing VDAs through Active Directory Group Policy using individual MSIs is not recommended and might fail. Citrix recommends using the startup scripts provided on the XenDesktop 7 installation media, as described in Install or remove Virtual Delivery Agents using scripts in Active Directory. [#383432, #372136]

• When upgrading a VDA using RDP, an error message titled "picaDispMgr.exe - Entry Point Not Found" might appear
during the Virtual Desktop Agent upgrade task. No action is needed. Click OK to dismiss the dialog box, and continue with the upgrade. [#376345]

- After upgrade, all Delivery Groups are accessible to every user, although the user is not member of the Delivery Group. Use the following command to set appropriate Delivery Group permissions:


If you made any manual modifications that specifically change the value of the AllowedUsers property of an Access policy rule, this command removes them. You must make modifications to remove the Access policy rules you do not want to change. [#0401360]

- The optimization phase of the Virtual Delivery Agent (VDA) installation might take a long time to complete. In some tests, it has taken about half an hour and may take longer. These instances occur when installing the VDA on an image running a Windows operating system if, on the XenDesktop Installation wizard Features page, Optimize Performance has been selected. This causes the Microsoft Native Image Generator (Ngen) to run. If this occurs, allow the installation process to finish. Citrix recommends that you run Ngen on your VDA base image prior to provisioning virtual desktops to avoid delays caused when it runs in the background of provisioned virtual desktops. [#381437]

- Creating a machine catalog fails in an environment running VMM 2012 SP1 and Hyper-V 2012 when using a ODX-enabled Storage Area Network. Follow the instructions in http://support.citrix.com/article/CTX139333 to resolve this issue. [#424040]

- During upgrade, if an error message mentioning PICAIsPorticaV2 entry point not found appears during an upgrade, it can be safely ignored. Complete the upgrade process and restart the machine when prompted. [#423947]

- When installing XenDesktop 7.1, you might encounter the following issue: The Virtual Delivery Agent (VDA) for workstation operating systems cannot be installed on evaluation versions of Windows 8.x. The Installation Options screen displays the message “Cannot be installed on this operating system.” The issue occurs because the XenDesktop installation program is incorrectly identifying Windows 8 evaluation versions as unsupported operating systems. A hotfix that addresses the issue is available as Knowledge Center article CTX139660. The hotfix lets you patch the installer before running it on Windows evaluation versions.

- When upgrading from XenDesktop 5 to XenDesktop 7.1 or XenDesktop 7.0, a hosted application assigned to both Shared and Private Delivery groups produces a Delivery Group incompatibility error during upgrade. To avoid this error, do not assign hosted applications to both Shared and Private Delivery Groups. [#419424]

- After upgrading from XenDesktop 7.0 to XenDesktop 7.1, when you select Test Site from the Studio Common Tasks pane, the following error appears:

  Controller <DNS name of Controller>: Service instance http://<DNS name of Controller>/Citrix/EnvTest/PeerAPI/v1 published by this service is not registered with the Configuration service. Register all service instances published by this service with the Configuration Service.

  Resolve this issue by running the following PowerShell on any Controller in the Site:

  Add-PsSnapIn Citrix.Configuration.Admin.V2

  Add-PsSnapIn Citrix.EnvTest.Admin.V1

  Get-EnvTestService | ForEach-Object { Get-EnvTestServiceInstance -AdminAddress $_.DNSAddress | Where-Object { $_.InterfaceType -eq "Peer" } | Register-ConfigServiceInstance } [#429469]

- After upgrading from XenDesktop 7.0 to XenDesktop 7.1, when you select Test Site from the Studio Common Tasks pane errors relating to the Monitor Service database schema appear. You can safely ignore these errors, as the Site is not affected. Do not restore your database from the most recent backup if the following errors are the only errors you see in the report.
StoredProcedure MonitorData.DeleteMachinesByCatalogId object definition does not match the expected reference schema.

Restore the database <name of SQL instance>\<Monitor database name> to the most recent backup.

StoredProcedure MonitorData.DeregisterMachinesByCatalogId object definition does not match the expected reference schema.

Restore the database <name of SQL instance>\<Monitor database name> to the most recent backup.

StoredProcedure MonitorData.RemoveMachinesByDesktopGroupId object definition does not match the expected reference schema.

Restore the database <name of SQL instance>\<Monitor database name> to the most recent backup.

StoredProcedure MonitorData.TerminateSessionsBySid object definition does not match the expected reference schema.

Restore the database <name of SQL instance>\<Monitor database name> to the most recent backup.

[#429471]  
[#436033]

Studio issues

- Studio startup is delayed if the machine where it is installed does not have an Internet connection. (For security, Citrix signs .NET-based components with an Authenticode signature; Studio startup is delayed if Windows cannot verify the signature.) As a workaround, disable the Authenticode signature checking feature as described in http://support.citrix.com/article/CTX120115; [#337698]
- When using Machine Creation Services (MCS) to provision machines on VMware vSphere, the combination of CPU sockets and cores on the provisioned machines reflects the combination of CPU sockets and cores on the base image used to create the machine catalogs. During MCS catalog creation, if the number of Virtual CPUs selected is higher than the maximum possible on the host, then provisioned machines are created with the maximum possible sockets and cores for that host, without indication to the user. [#331269]
- When using System Center Virtual Machine Manager in a pure IPv6 environment, and using Machine Creation Services to create machine catalogs, all VMs have both IPv4 and IPv6, even if the master VM is configured without IPv4 in the TCP/IP stack. This is a third party issue with Microsoft, and there is no workaround. [#371712]
- When using VMware vSphere in an IPv4 and IPv6 environment with VMware ESX hypervisors configured with VMXNET3 network adapters, all VMs have both IPv4 and IPv6, even if the master VM is configured without IPv4 in the TCP/IP stack. This is a third-party issue with VMware, and there is no workaround. [#371712]
- Long machine catalog names and long storage path names might cause a disk attach error in the VMM job window. Microsoft has identified a maximum limit of 255 characters on the length of the file path for VM resources. This issue has been seen when using local storage on a standalone Hyper-V host, due to the long file path used to store the VMs; however, it is not limited standalone Hyper-V hosts. [#359673]

As a workaround:

- Create VMM MCS catalogs with short names, especially when the storage is accessed using a long path.
- Shorten the path to the storage used to store the VMs.
- Although this release supports cross-forest trusts, registration for MCS VMs works only if reverse lookup zones are added to the Domain Name System (DNS) environment. [#338043, 342432]
- To change a database to one that was previously used, you must use the SDK; it cannot be done from Studio. [#355993]
To switch to the Configuration Logging database:

- `Set-LogDbConnection -DataStore 'Logging' -DBConnection $null`
- `Set-LogDbConnection -DataStore 'Logging' -DBConnection '<new database connection string>'`

To switch to the monitoring secondary database:

- `Set-MonitorDbConnection –DataStore 'Monitor' –DBConnection $null`
- `Set-MonitorDbConnection –DataStore 'Monitor' –DBConnection '<new database connection string>'`

For example, the new database connection string could be:

'Server=dbserver;Initial Catalog = dbname; Integrated Security = True'

When you use MCS to provision machines on your hypervisor platform, CPUs are added but no cores. If you change the CPU value during catalog creation, the products licensed on a per CPU basis might need more licenses. For example, your master image has one CPU and four cores and you change the CPU value during catalog creation.

If, during MCS catalog creation, you select more CPUs than the maximum possible on the host, provisioned machines have the maximum number supported for that host and you are not notified of the difference. For example, a Desktop OS machine can use only two physical CPUs; thus, you will see only two even if more are assigned.

As a workaround, ensure your master image VM has the same virtual CPU configuration that you want to deploy for the catalog. [#331274]

- Microsoft does not support Lync 2013 as a hosted application on Server OS machines. Attempting to launch it from the server causes the server to fail. This is a third party issue and there is no workaround. With Receiver 4.0, Lync can be launched successfully from Desktop OS machines [VDI workers] [#371818]
- Studio might incorrectly display the correct status of long-running tasks. (For example, failed tasks show as in-progress.) If you know the task completed, ignore the display. However, do not restart Studio if a long-running task is active; the task might not complete correctly. [#311185, #335709]
- When launching a seamless application on a Windows server, the window might not have the Aero theme, even when Enhanced Desktop Experience is enabled.
  - Users launching only seamless applications never get the Aero theme.
  - Users with a mix of seamless applications and desktops get the Aero theme after establishing the first desktop session; then, seamless applications have the Aero theme.
  - As a workaround, set the Citrix Enhanced Desktop theme in the default user profile; all users on that VDA get the Enhanced Desktop Experience for their seamless applications. The theme is part of the VDA install and must be set for all VDAs. [#348812]
- Studio messages sent to a Windows 8 machine will display on the Windows 8 Desktop, and not the Windows default (formerly called Metro) display mode. The user must switch to Desktop mode to see the message. This is a third party issue and there is no workaround. [#387356]
- If a Site contains more than one hosting infrastructure object with the same name, the Studio display might not be correct. When you create hosting infrastructure objects (for example, networks and storage), it is best practice to specify a unique name for each. [#384959]
- After using Delegated Administration to create an administrator with a new scope, refresh the Studio display. Otherwise, you might receive a permissions error when you log on as the new administrator and attempt to create a new connection or resource. [#386634]
- When you use the PowerShell SDK to change a Site name (set-configsite -sitename "change name"), you will receive an error if you attempt to join another Controller to the Site. As a workaround, rename the Site to its original name (to match the ServiceGroupName on the output of Get-ConfigRegisteredServiceInstance) before joining the new Controller.
to the Site. After joining the Site, you can rename the Site again. [#386919]

- When connecting to a hosted application such as Notepad from a random Delivery Group, the application initially begins to open. However, after the loading dialog box disappears, the application fails to appear. The issue occurs when the default one-minute time-out is exceeded and the session exits automatically because the application takes a longer time to start. [#389025]

As Administrator, change the base image and re-provision with a changed timeout value as follows:

Locate the registry key: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Citrix\wfshell\TWI (Create the key if it is not available.)

Name: ApplicationLaunchWaitTimeoutMS

Type: REG_DWORD

Data: Required additional time-out, in milliseconds

Note: Specifying a value of less than 10000 reverts to 10000 because 10 seconds is the minimum override.

- When XenServer hosts are in a pool and a XenDesktop secure connection to the pool occurs, you may receive the following error message: Invalid Connection Address. Check that the address is valid and that it references a host in the XenServer pool.

To avoid this scenario, install certificates issued by a trusted root certificate authority on all the XenServer hosts in the pool. [#392279]

- When creating a machine catalog through MCS, a Database failover might occur when copying the master image or when adding a machine. The following message appears: Collection was modified; enumeration operation may not execute. Make sure that the database is in a healthy state before performing the following actions. If the failover occurs:

During the master image copy, delete the machine catalog and create it again. When adding machines, select the MCS machine catalog and click Add Machines. If an error persists, delete the machine catalog and create it again. [#393889]

- If the Citrix Licensing service on a License Server is not running when you click on the Licensing node in Studio, the following error message appears: “Object reference not set to an instance of an object.” To prevent this issue, make sure that the Citrix Licensing service is running before attempting to connect to it through Studio. [#389439]

- To edit Host connections in Studio, you must use the Full Administrator role or the Machine Catalog Administrator role and the All scope. Without the All scope you may experience an unexpected error. [#392799]

- In some instances, if you delete machine catalogs and then attempt to delete the Hosting Unit, you might receive a message indicating that the resources are still in use. This issue is caused by an internal task that could take a significant time (up to eight hours) to complete. If you repeat the Hosting Unit deletion after the task has completed, the delete is successful. [#392462]

- You cannot remove a Delivery Controller that is part of a partially upgraded Site. As a workaround, complete the following steps:

  1. Upgrade one or more (but not all) Delivery Controllers.
  2. Using the upgraded Delivery Controller, run Studio to perform an in-place upgrade of the Site Database.
  3. Upgrade all remaining Delivery Controllers.

You can now remove the Delivery Controller of your choice. [#384601]

- When adding or modifying Virtual Machine or Personal vDisk storage for a host on the Storage page, storages that are part of the hosting unit might be greyed out. In addition, you cannot uncheck the storage and supersede the storage. Therefore, the Storage status does not change to "Not accepting new machines" as stated on the Standard and PVD
As a workaround, use the PowerShell Set-HypHostingUnitStorage with the -Superseded parameter to supersede the storage, as shown in the following example:

```
Set-HypHostingUnitStorage -LiteralPath XDHyp:\HostingUnits\MyHostingUnit
-StoragePath 'XDHyp:\HostingUnits\MyHostingUnits\newStorage.storage'
-Superseded $true
```

- When updating a machine catalog, although the update is successful, notification messages may fail to appear on sessions running on machines in this catalog. As a workaround, the administrator can attempt to send the message again through the Search node in Studio. [#390726]

- To discover and set up applications in Delivery Groups, make sure that there are some machines that are registered and can be powered on. For App-V application to be discovered, you need to configure App-V Publishing in Studio. [#393676]

- After a user launches the first App-V application, launching a second App-V application too quickly might fail. If this occurs, the user should wait a few minutes so that the initial synchronization can complete, and then launch the second application again. For details about this issue, see CTX138056. [#397521]

- If you change either the Configuration Logging or Monitoring database in Studio, a previously created mirror database does not correctly set the failover partner. To correctly set the failover partners for mirrored databases, use the PowerShell Set-LogDataStore cmdlet for the Configuration Logging database and Set-MonitorDataStore for the Monitoring database. [#371661]

- When a physical machine is newly configured for use with Remote PC Access, the first attempt to register with a XenDesktop Site might fail. After about five minutes, the machine successfully re-registers. Prior to this successful re-registration you cannot connect to the machine using StoreFront. [#394791]

- VMware vSphere 4.1 is not supported. See System requirements for XenDesktop 7.1 for supported versions. [#397658]

- If you attempt to create secure connection in Studio to a vCenter on a Delivery Controller where the vCenter certificate is not installed, Studio fails to warn that the connection will not function correctly unless the vCenter certificate is installed on all Controllers in the site. To prevent this issue, install the vCenter certificate on all Controllers. [#399201]

- If you attempt to create a connection in Studio to a Microsoft System Center Virtual Machine Manager (VMM) on a Delivery Controller where the VMM Console is installed, Studio fails to warn that the connection will not function correctly unless VMM Console is installed on all Controllers in the site. To prevent this issue, install VMM Console on all Controllers and restart the Controllers. [#348614]

- If the antivirus program BitDefender is installed on a VDA, you might not be able to create machine catalogs. There is no work-around for this issue. [#392705]

- When XenServer 6.2 or 6.1 is configured with an open vSwitch controller, MCS will fail to create machines unless the option The Master Image is built using Windows XP or Windows Vista is checked. Selecting this option means that some XenDesktop 7.0 features are unavailable for this catalog. [#394771]

- In Studio, when selecting one or more file type associations for an application, clicking Apply and then clicking OK does not save the association. Implement associations by clicking only OK. [#404756]

- XenDesktop 7 does not support creating resources to vSphere resource pools. Citrix does not recommend upgrading to or deploying XenDesktop 7 if you are currently using resource pools. Doing so causes failure when creating or updating catalogs. [#401358]

- Although an administrator's role and scope include permission to enable or disable maintenance mode on a Host, a permission error might occur when attempting either action. As a workaround, be sure the role and scope include permission to edit Host Connection properties. [#418506]

**Director issues**

- When monitoring Windows XP virtual desktops running WinRM 2.0 for users with VDAs earlier than 7, you must change the WinRM port listening order. For details about how to change the setting to 5985,80, see Advanced configuration.
When using Internet Explorer 10, you must use the recommended browser settings; otherwise, you cannot access Director. When you install Internet Explorer 10, accept the default to use the recommended security and compatibility settings. If you already installed the browser and chose not to use the recommended settings, go to Tools > Internet Options > Advanced > Reset and follow the instructions.

Details about a user’s session do not appear in the Help Desk and User Details views of Director if the Fast User Switching feature disconnected the session due to another user logging on to the same machine. If this issue occurs, ask the first user to restart the VDA.

When you use Firefox to search for a user name in Director by typing Chinese characters, the auto-complete functionality might fail. Use one of the following workarounds:

- Type the full user logon name.
- Type the user name, and then press the space bar.
- User an Internet Explorer or Chrome browser instead of Firefox.

The total session count in the Dashboard view is sometimes higher than the session count in other views because it contains historical data. Other views, such as the Filters view, contain real-time data.

Logon duration does not update in Director’s Dashboard view if users launch a hosted application.

Using Active Directory Users and Computers to configure users with logon scripts fails and data does not appear in the Logon Duration panel. Instead, to configure users with logon scripts, you must use a Group Policy.

The logon duration data from a first-time logon to a Personal vDisk VDA might not be collected or displayed in Director. For subsequent logons, data appears normally.

By default, Director utilizes Global Catalogs for Active Directory searches which cause long user search times if any of the domains cannot be contacted. Refer to CTX133013 for more information to prevent this issue.

Director logins may fail and result in a reloaded log in page when the Director server is installed on Windows Server 2008 R2 and Director is accessed from Internet Explorer 11. Make the following changes to the Web.config file:

```xml
<forms protection="All" timeout="240" name="DESKTOPDIRECTORAUTH" path="/" requireSSL="false" slidingExpiration="true" defaultUrl="default.html" loginUrl="LogOn.aspx"> with <forms protection="All" timeout="240" name="DESKTOPDIRECTORAUTH" path="/" requireSSL="false" slidingExpiration="true" defaultUrl="default.html" loginUrl="LogOn.aspx" cookieless="UseCookies">
```

As a workaround, register the VDA, log off the current session, and log in to the VDA again.

When you run a console session from a Windows Server 2012 desktop, navigating to the user details page in Director might result in an error and no data will be displayed. As a workaround, register the VDA, log off the current session, and log in to the VDA again.

HDX issues

Note: For the latest updates to HDX Flash compatibility, refer to CTX136588.

- With GPU pass-through and NVIDIA Kepler cards, the first connection attempt may fail for a HDX 3D Pro user device with three or four monitors. If this happens, attempt to connect again.
- Even though webcams might support H.264 compression, this release does not support hardware compression, so you must use software compression. To do this, add a registry entry on user devices at HKEY_CURRENT_USER\Software\Citrix\HdxRealTime; add a DWORD registry name DeepCompress_ForceSWEncode. When set to 1, software compression is used. By default, this setting is off and hardware compression is used.
HDX RealTime Webcam video lags if the video resolution is higher than 720p (1280x720). [350187]
When using HDX Flash Redirection continuously, the session might become unresponsive. [350085, 361926]
HDX RealTime Webcam supports most of raw formats supported by a webcam, but in rare cases, if the webcam has an unsupported format, that webcam might not work as the Citrix HDX Webcam. [338318]
Multiple duplicate images might be seen intermittently when using Citrix HDX Webcam with some models of webcams. [367322]
HDX RealTime Webcams are not supported for these applications:
- Citrix GoToMeeting when hosted on Server OS Machines with Windows 2012 operating systems. [346430]
- GoToMeeting (on any platform) if the webcam is attached after a meeting has started. [346140]
- Skype 6.0 or later. This is a third-party issue. [349545]
- Microsoft Lync 2013 and Adobe Connect with VDAs on Windows 8 and Windows 2012 operating systems. [340784, 348506]
Microsoft Office Communications Server (OCS) video calls if the Webcam is attached after the call is in progress. [370236]
Microsoft Silverlight. This is an intermittent issue. As a workaround, on the user device, enable the legacy codec by adding a DWORD registry key value name EnableDeepcompress_Client at HKEY_CURRENT_USER\Software\Citrix\HdxRealTime and setting it to 0. [379779]
- 64-bit video conferencing applications. Video compression for 64-bit applications is not supported. [366515]
When using HDX 3D Pro with the XenDesktop 5.6 Feature Pack 1 Virtual Desktop Agent on Windows XP virtual desktops, during the first connection, the Fine Drawing (2D) check box is selected and sometimes greyed out. This is due to delayed registry updates, which cause the Config tool UI to pick up incorrect default values during initialization. [353031]
As a workaround, disconnect and reconnect the session.
If the problem persists, clear previous session information by deleting settings under the following registry entry:
HKey_Current_Users\Software\Citrix\HDX3D\BitmapRemotingConfig
When viewing the Display Adapters node from the Device Manager console applet, the Standard VGA Graphics Adapter appears in the list with a yellow exclamation point (yellow bang). You can ignore this warning because it does not affect functionality. This warning occurs because a legacy model XPDM display driver (Standard VGA Graphics Adapter) is not allowed to load when a new model WDDM display driver (Citrix Display Driver) is installed. [339390]
Users might experience issues when attempting to play media files on Windows 8 user devices. This is because XenDesktop fails to register the correct default program for client-side content fetching protocols used to stream media files to user devices. As a workaround for Microsoft Media Streaming (MMS) and Real Time Streaming (RTS) protocols, change the default program used for playing media files from Windows Media Player to Citrix CSF Handler. There is no workaround for the Hypertext Transfer Protocol (HTTP). [328805]
TWAIN redirection fails on hosted shared desktops and applications. This is a third party issue related to TWAIN applications that require TWAIN binaries to be located in certain paths. [300854, 340999]
As a workaround, on your Windows Server 2012 machine running the VDA, copy these files to the following locations:
- Copy "twain_32.dll" to the \"\WINDOWS\" directory of the User profile (for example, copy twain_32.dll into the folder: "%USERPROFILE%\Windows\"").
- Copy "twain_32.dll.mui" into the \"\WINDOWS\en-US\" directory of the User profile (for example, copy
The 64-bit Windows Media Player or QuickTime player cannot play some video files using server-side rendering when HDX MediaStream Windows Media Redirection is disabled. As a workaround, use the 32-bit version of Windows Media Player. [#384759]

Universal Print Server printers selected in the virtual desktop do not appear in the Devices and Printers window in Windows Control Panel. However, when users are working in applications, they can print using those printers. This issue occurs only on Windows Server 2012 and Windows 8 platforms. [#335153]

With GPU pass-through and NVIDIA Kepler cards, the first connection attempt may fail for a HDX 3D Pro user device with three or four monitors. The second connection should be successful. [#422049]

HDX Monitor 3.1 displays the message "Can't connect to target machine" when you start the tool on a Windows 8.1, Windows 8, or Windows 7 VDA. To work around this issue, open a command window on the VDA and run the following command as an administrator:

```
winrm quickconfig
```

Then, restart HDX Monitor [#404817]

The user's Windows computer stops responding when a GoToMeeting session using a webcam configured for USB redirection is started in a Remote PC Access session with an Intel Core i7 processor-based computer. If this occurs, restart the user's computer and restart the Remote PC Access session. The session resumes where the disconnection occurred. To avoid this occurrence, use HDX webcam video compression instead of USB redirection. For details, see http://support.citrix.com/proddocs/topic/xendesktop-7/hd-new-graphics-video.html [#423284]

User devices running Receiver for HTML5 might be unable to connect to a Server OS machine running Windows Server 2012 R2. To avoid this issue:

- Use an existing machine, rather than a machine created with Machine Creation Services (MCS) or Provisioning Services, as the Windows Server OS machine.
- If you plan to use machines created with MCS, on the master image for the catalog, edit registry and create the following DWORD value:

```plaintext
HKEY_LOCAL_MACHINE\Software\Citrix\GroupPolicy\Defaults\IcaPolicies\AcceptWebSocketsConnections = 1
HKEY_LOCAL_MACHINE\Software\Citrix\GroupPolicy\Defaults\IcaPolicies\AllowDesktopLaunchForNonAdmins = 1
HKEY_LOCAL_MACHINE\Software\Citrix\GroupPolicy\Defaults\IcaPolicies\WebSocketsPort = 8008
```

Because the WebSockets port number is set by editing the registry, it is not necessary to enable the Websockets connections Citrix policy for the catalog.

- If you are using a machine created with Provisioning Services, follow the recommendations in CTX139265. [#424064]
- GRID enabled 3DPro VDI does not support apps that change resolution in full screen. As a workaround, launch full screen apps with Client Screen Resolution. [#444793]

Licensing issues

- When licensing is uninstalled and then reinstalled and a read-only XenDesktop administrator attempts to view Licensing information in Studio, Studio displays the error "You do not have permissions to perform this operation." A read-only administrator does not have the permission to trust a new License Server. As a workaround, a full license administrator must go to the Licensing node and authenticate the License Server. [#380982]
- If you installed the License Server without successfully configuring it (with the post installation License Server
Configuration tool), any subsequent License Server upgrade fails. As a workaround, ensure that every License Server installation is configured with the post-installation License Server Configuration tool. [#377079]

- When you try to start the License Administration Console or the Simple License Service, a blank page might display if the Internet Explorer Enhanced Security Configuration is enabled and the License Administration Console or the Simple License Service is not in the Trusted Sites zone. Workaround: Disable Internet Explorer Enhanced Security Configuration. [#382429]

- If port 8083 is in use when you install or upgrade XenDesktop, the License Server configuration and XenDesktop installation fail with a License Server Configuration Failed error. As a workaround, check the event log to ensure the error is actually "Port in use." If it is:
  1. Uninstall the License Server by double-clicking on the CTX_Licensing.msi in the x64\Licensing folder on the XenDesktop media.
  2. Run the installer again. It displays some components as Partially installed. Click Install and the installer completes the installation and configures any necessary XenDesktop components.
  3. Manually install the License Server and specify port numbers that do not conflict with other applications on the machine. [#390815]

- The user list for the License Administration Console and the Citrix Simple License Service Web page does not support non-ASCII characters in user/group names. Due to this limitation, on a Russian operating system, the BUILTIN Administrators group is not added to the user list because it is created with non-ASCII characters. This issue applies to both fresh installs and upgrades. Any users belonging to the BUILTIN Administrators group in an earlier release of XenDesktop and the Simple License Service will not have access to the License Administration Console or the Simple License Service after an upgrade. As a workaround, add ASCII-character versions of Russian users/groups names post-installation using the License Administration Console interface. Alternatively, install the license server on one of the other supported operating systems. [#395305]

- When you install or upgrade XenDesktop and you have Perpetual (permanent) licenses, Studio might display your licenses with an expiration date of 01/01/2000. You can ignore this expiration date and launch your desktops and applications. [#402975]

- You cannot use Studio to configure your site to use your XenApp Platinum licenses. If you select XenDesktop Apps Edition and Concurrent Licensing in the Studio Edit product edition UI, the site is configured to consume XenApp Enterprise licenses. Workaround: Use the PowerShell Set-ConfigSite cmdlet:
  Set-ConfigSite -ProductCode MPS -ProductEdition PLT -LicensingModel Concurrent

Local App Access issues

- In a Windows 8 or Windows Server 2012 hosting environment, if Local App Access is enabled and the extension for a client hosted app does not have a File Type Association (FTA), FTA redirection fails. The user is prompted to select “Look for an app in the Store” or “More options.” [#372834]
  As a workaround, use one of these methods on the VDA master image:
  - Rename the DelegateExecute registry value for HKEY_CLASSES_ROOT\Unknown\shell\openas\command\DelegateExecute.
  - Use Notepad to open a file with the extension. FTA redirection will work for subsequent attempts.
  - URL redirection is disabled, by default, by Microsoft on Windows Server 2012. To enable it, disable Internet Explorer enhanced configuration mode. [#356260]
  - Changes to Local App Access properties during a session do not take effect automatically. As a workaround, log off and log back on. [#357488]
  - When publishing a local application in Studio on Windows Server 2012 or Windows 8, the OK button on the Select icon
page is inactive after you browse to a DLL and select an application icon. To work around this issue, use Studio on Windows Server 2008 R2 or Windows 7. You can also work around this issue on Windows Server 2012 or Windows 8 as follows: After you browse to a DLL, click Choose from Citrix default icons, then click Choose an icon from a file on a representative machine, select the icon, and click OK. [#403710]

- If Local App Access applications are launched on Windows 8 or Windows 2012 platforms, those VDAs cannot be launched from the Modern shell. As a workaround, close the local application and then launch the VDA application. [#359670]

- Shellhook.dll is not loaded with Receiver for Windows 3.4 and earlier when local applications are launched. As a workaround, change the value of the registry key LocalApInit_DLLs to 1 under HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows. If the workaround is not used, client to server FTA redirection does not work correctly. [#356130]

- URL redirection might fail for Internet sites that have a pop-up blocker enabled. As a workaround, disable the pop-up blocker. However, for security reasons, disabling pop-up blocker is not recommended. [#371220]

- When the Aero mode is enabled for a VDA session, there are inconsistencies between the VDA local applications and the client-hosted applications (for example, ALT+Tab; flashing taskbar entries, jump list, live preview). For compatibility, disable the Aero mode. [#361043]

- Flash Redirection has compatibility issues such as a WMP black screen and flash pseudo-container window out of bounds. As a workaround, disable Flash redirection. [#360182]

- The Desktop Composition Redirection graphics feature is disabled when Local App Access is enabled. To use that feature, disable Local App Access. [#377386]

- With Session Reliability and Local App Access enabled, if network connectivity is disrupted after you click Home in the XenDesktop toolbar and display the client’s desktop, the VDA session might not display the last screen shown before the network disruption. Instead, only the XenDesktop toolbar and the client’s desktop appear. On the toolbar, only the Disconnect option works. [#357769]

- When Local App Access is enabled and a user changes the XenDesktop session to full screen before or while playing a media file, some or all of the image does not appear. To work around this issue, relaunch the XenDesktop session. [#402702]

- Launching client-hosted shortcuts in the Local Programs folder in a session connected from Japanese or Chinese clients might result in an application error. To resolve this issue, log off from the session then launch the session again. If the problem persists, create a client-hosted application as described in Provide access to local applications to publish specific local applications in place of client-hosted shortcuts [#399100]

Personal vDisk issues

- After upgrading the VDA from XenDesktop 5.6 or XenDesktop 5.6 Feature Pack 1 to XenDesktop 7, the Citrix Personal vDisk dialog box does not appear after you modify a master image. You are therefore not reminded to update the Personal vDisk inventory. As a workaround, click Update Inventory on the Start menu. [#394892]

- The presence of antivirus products can affect how long it takes to run the inventory or perform an update. Performance can improve if you add CtxPvD.exe and CtxPvDSvc.exe to the PROCESS exclusion list of your antivirus product. These files are located in C:\Program Files\Citrix\personal vDisk\bin. [#326735]

- Symantec Endpoint Protection 12.1.2 LiveUpdate feature does not work with personal vDisk-based pooled desktops. Additionally, the Scan for Threats and View Quarantine features are not available. This is a third-party issue. [#380622]

- A personal vDisk may fail to start if an Update Sequence Number (USN) journal overflow occurred due to a large number of changes made to the system after inventory update. To avoid this, increase the USN journal size to a minimum of 32 MB in the master image and perform an image update. [#369846]

- If copy-on-write is enabled and alternate data streams are used, changes made to the contents of the alternate data streams are not saved. Copy-on-write does not support alternate data streams. To avoid this conflict, relocate the files...
containing alternate data streams to the guest volume. For details, see the documentation in the custom rule files in %ProgramData%\Citrix\personal vDisk\Config with custom_* names. [#369193]

- Hard links between files inherited from the master image are not preserved in personal vDisk catalogs. [#0368678]
- After upgrading from Office 2010 to 2013 on the Personal vDisk master image, Office might fail to launch on virtual machines because the Office KMS licensing product key was removed during the upgrade. As a workaround, uninstall Office 2010 and reinstall Office 2013 on the master image. [#391225]
- Personal vDisk catalogs do not support VMware Paravirtual SCSI (PVSCSI) controllers. To prevent this issue, use the default controller. [#394039]

For virtual desktops that were created with Personal vDisk version 5.6.0 and are upgraded to 7, users who logged on to the master virtual machine (VM) previously might not find all their files in their pooled VM. This issue occurs because a new user profile is created when they log on to their pooled VM. There is no workaround for this issue. [#392459]

- Personal vDisks running Windows 8 may experience multiple Security-SPP entries in their event logs. [#0353437]
- Personal vDisks running Windows 7 cannot use the Backup and Restore feature when the Windows system protection feature is enabled. If system protection is disabled, the user profile is backed up, but the userdata.v2.vhd file is not. Citrix recommends disabling system protection and using Backup and Restore to back up the user profile. [#360582]
- When working with personal vDisks, applications that depend on User Account Control (UAC) virtualization may not work. The Virtual File Store will not be available to these applications. [#351900]
- When you create a VHD file on the base VM using the Disk Management tool, you might be unable to mount the VHD. As a workaround, copy the VHD to the PvD volume. [#355576]
- Sometimes, after performing an image update on a virtual desktop, the personal vDisk associated with the desktop does not start and an error (error code 0x10) is displayed. There is no workaround for this issue. [#416128]
- Office 2010 shortcuts remain on virtual desktops after this software is removed. To work around this issue, delete the shortcuts. [#402889]
- When using Microsoft Hyper-V, you cannot create a catalog of machines with personal vDisks when the machines are stored locally and the vDisks are stored on Cluster Shared Volumes (CSVs); catalog creation fails with an error. To work around this issue, use an alternative storage setup for the vDisks. [#423969]
- When you log on for the first time to a virtual desktop that is created from a Provisioning Services catalog, the desktop prompts for a restart if the personal vDisk has been reset (using the command ctxpvd.exe -s reset). To work around this issue, restart the desktop as prompted. This is a once-only reset that is not required when you log on again. [#340186]
- If you install .NET 4.5 on a personal vDisk and a later image update installs or modifies .NET 4.0, applications that are dependent on .NET 4.5 fail. To work around this issue, distribute .NET 4.5 from the base image as an image update. 
- Upgrading to Symantec Endpoint Protection 12.1.3 through an image update causes symhelp.exe to report corrupt antivirus definitions. There is no workaround for this issue. [#423429]
- On Windows 8.1 desktops in a XenServer 6.2 deployment, image updates of personal vDisks might fail with status code “7 Error Code: 0x6”. The Personal vDisk logs indicate that RingCube.dat could not be mounted. This is due to an incompatibility of Windows 8.1 with XenServer 6.2. To work around this issue, use a version of XenServer that fully supports Windows 8.1. [#413384]

Desktop Lock issues

- After installing Citrix Desktop Lock on a domain-joined Windows 8 machine and restarting it, the desktop background may become black and a My Computer window may be displayed instead of the usual Start screen. To log off the machine, type logoff in the address bar of the window, and press Enter. For more information about configuring Windows 8 machines with Desktop Lock, see Configure the Desktop Lock. [#329075]
- If a Windows 8 machine fails and the user attempts to restart it by pressing Ctrl+Alt+Del, the Windows Security dialog on the local desktop opens. This issue occurs even when the Citrix Desktop Lock software is installed and configured. To resolve this issue, select Start Task Manager to display the Restart dialog. [#337507]
The Citrix Desktop Lock does not redirect Adobe Flash content to domain-joined user devices. The content can be viewed but is rendered on the server, not locally. As a workaround, configure Adobe Flash redirection for server-side content fetching to pass the content from the server to the user device. This issue does not occur on non-domain-joined devices or when the content is viewed with the Desktop Viewer. [#263092]

Failure to start the virtual desktop agent (VDA) on the user device may occur if the device is running Windows 7 with Desktop Lock and Receiver for Windows Enterprise 3.4. This may occur if the user is or is not connected to the Internet. The error message, “No Internet Connectivity,” may appear when this occurs. This is a third party issue with Microsoft. For a potential resolution, see http://technet.microsoft.com/en-us/library/cc766017. [#408642]

End-user and VDA issues

- Handle leaks by wfica32.exe can occur on the user device when playing Windows Media Player files continuously in a Windows 32-bit XP client session. [#378146]

- When using Windows Media Player on a Windows 8 machine with a VDA that uses MMS, successive files might not play. As a workaround, close and restart Windows Media Player. [289797]

- Disconnect button is not available to users. As a workaround, provide a shortcut to the TSDiscon.exe utility, which is included with the operating system; this will allow them to disconnect from sessions. [#362937]

- Starting a VDA from a user device with a smart card reader might fail if the user previously started the VDA from that device and selected Disconnect from the Desktop Viewer. In this case, the user might see the smart card credential screen or the message ‘Reading smart card.’ As a workaround, choose one of the following:
  - Remove and reinsert the smart card in the reader
  - Click Cancel. Then, in the VDA, press Ctrl+Alt+Del. [#322301]

- The user may continue to have access to the remote PC after attempting to disconnect. This occurs when the user selects the Disconnect command from the Start menu. There is a delay after Disconnect is selected. If the user presses Ctrl+Alt+Del during this delay, the VDA on the remote PC remains available for several minutes before disconnecting. During that time, the VDA on the user device freezes until the remote VDA disconnects. [#322301]

- For VDAs installed on Windows 7 and Windows 8 platforms, two mouse pointers might be visible: one movable and one locked to the UI. This is a third party issue with the Nvidia driver. As a workaround, you can disable Nvidia GRID technology (formerly known as VGX) by running MontereyEnable.exe -disable -reset, and then restarting the machine. [#307921]

- On Windows 7 machines, custom resolutions or ICA connections fail if users have an HDX 3D Pro physical headless VDA. This is a third party issue because the Microsoft default virtual monitor is incompatible with the Nvidia card. As a workaround, edit the registry, under HKLM\Software\Citrix\HDX3D\BitmapRemotingConfig, create the registry key HKLM_ForceVirtualMonitors, and set the value to 1. This change forces the creation of two virtual monitors and resolves the issue. [#380060]

- The Microsoft Desktop Composition feature has a scalability cost for XenDesktop VDAs. For users requiring maximum scalability, Desktop Composition should be disabled by Microsoft policy for any user not using Desktop Composition Redirection. [#386602]

- For Remote PC Access connections, users with dual monitors might experience an intermittent issue where windows, icons, or the task bar display on the wrong monitor on the local console after disconnecting from a remote session. [#364575]

- In certain scenarios, when users attempt to unlock a locked session locally, the session relocks itself repeatedly. This issue may occur if the user unplugs or powers off a keyboard locally connected to a Remote PC Access machine during a remote ICA session. As a workaround, users should relaunch the session remotely, disconnect the session, and then log on again from the console. [#382554]

- When installation of a VDA for Windows Server OS from the graphical interface appears to hang, check for an error message (“Printers - The arguments are invalid”) which might appear behind the main installation window. This message
appears if the Print Spooler Service has not yet started. Either wait for the Print Spooler Service to start, or start the service manually. After clicking OK in the message box, installation continues. [#385526]

- For VDAs earlier than 7, users' data might not appear in Director even after you correctly configure Windows Remote Management (WinRM) for these VDAs. If this issue occurs, restart the WinRM service and the data should display in Director as expected. [#392047]

- In secure environments, a new VDA might fail to register with a Controller. Specifically, when installing a VDA containing a local security policy setting that allows only administrators to access a computer from the network, the VDA installs but cannot register with a Controller. Instead, a warning is issued that user access rights are not properly configured. For more information, see CTX117248. [#336203]

- Receiver for Windows users cannot log on to stores using pass-through authentication, even though the domain pass-through authentication method is enabled in the StoreFront authentication service. To resolve this issue, run the command Set-BrokerSite -TrustRequestsSentToTheXmlServicePort $True from a Windows PowerShell command prompt on the XenDesktop Controller. [#330775]

- ICA Roundtrip checks are not supported when legacy graphics mode has been specified using a policy for Windows Server 2012 VDAs. There is no workaround for this issue. Note that Windows 8 VDAs do not support legacy mode, so they are not affected by this issue. [#394824]

- When users connect directly to VDAs using RDP, the Controller sometimes mistakenly reports additional sessions that do not exist, and these non-existent sessions remain in a Connected state in Studio until the VDA is restarted. [#385823]

- If Client Drive Mapping is configured for Server OS and Desktop OS machines, edits made to existing files in applications might not be saved on the mapped drive and could show an error message. As a workaround, on the VDA machine, change the following settings to disable read and write caching. In this registry location: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\picadm\Parameters, change the following settings to 0: EnableCcReadCache=0, EnableCcWriteCache=0. As an immediate measure, users can save the file as a new name to preserve the changes in each file. [#395828]

- The Windows Connection Setting of a Virtual Delivery Agent (VDA) for Windows Server may report an incorrect value of LogonDisabled, even if the setting is enabled, causing sessions not to launch. This occurs when Allow Users to connect remotely using Remote Desktop Services is enabled for the VDA through Group Policy (Computer Configuration > Administrative Templates > Windows Components > Remote Desktop Services > Remote Desktop Session Host > Connections), yet Don't allow connections to this computer is selected in the Remote tab of the server's System Properties dialog box. Enabling Remote Desktop connections via Group Policy should override the disabling of Remote Desktop connections via a server's individual System settings. Instead, the server's individual System settings are taking precedence. For additional information, see http://support.microsoft.com/kb/2083411. [#402498]

- When using Machine Creation Services (MCS) to provision machines, only use ASCII characters for machine names. Using non-ASCII characters results in other unrelated characters for host names in provisioned machines. [#415134]

- When creating a Machine Creation Services (MCS) machine catalog, a name containing the back tick character (`) results in an error. Refrain from using this character in machine catalog names. [#414419]

- Users may encounter blank or black screens when logging in to Windows Server 2012 R2 machines, and be unable to connect to the server through RDP connection. In addition, on the server console some system processes such as task manager may also be unable to run, requiring a system cold boot.

To alleviate this problem, change the Enhanced Desktop Experience policy to Prohibited for Server 2012 R2 Delivery Groups. [#425255]
Components for this release

Jun 14, 2013

If you already have experience with a XenDesktop or XenApp environment, it will be helpful to you to identify the components introduced in this release and learn how they work and communicate with each other. Under the new architecture, XenDesktop and XenApp are unified, including management and delivery components, to give administrators a unified management experience.

This figure shows the key components in a typical deployment.

![Diagram of components](image)

The core components of this release are:

**Director** — Director is a web-based tool that enables IT support and help desk teams to monitor an environment, troubleshoot issues before they become system-critical, and perform support tasks for end users. You can also view and interact with a user's sessions using Microsoft Remote Assistance.

**Receiver** — Installed on user devices, Citrix Receiver provides users with quick, secure, self-service access to documents, applications, and desktops from any of the user's devices including smartphones, tablets, and PCs. Receiver provides on-demand access to Windows, Web, and Software as a Service (SaaS) applications.

**StoreFront** — StoreFront authenticates users to sites hosting resources and manages stores of desktops and applications that users access.

**Studio** — Studio is the management console that enables you to configure and manage your deployment, eliminating the need for separate management consoles for managing delivery of applications and desktops. Studio provides various
wizards to guide you through the process of setting up your environment, creating your workloads to host applications and desktops, and assigning applications and desktops to users.

License server — License server manages your product licenses. You must create at least one license server to store and manage your license files.

Delivery Controller — Installed on servers in the data center, the Delivery Controller consists of services that communicate with the hypervisor to distribute applications and desktops, authenticate and manage user access, and broker connections between users and their virtual desktops and applications. The Controller manages the state of the desktops, starting and stopping them based on demand and administrative configuration. In some editions, the Controller allows you to install Profile management to manage user personalization settings in virtualized or physical Windows environments. Each site has one or more Delivery Controllers.

XenServer — XenServer is an enterprise-class virtual machine infrastructure solution that creates the foundation for delivering virtual desktops and offers advanced management features. Multiple VMs can run on XenServer, which takes advantage of the advanced virtualization features of the latest virtualization-enabled processors from Intel and AMD. For more information about XenServer, see the XenServer documentation in eDocs.

Virtual Delivery Agent (VDA) — Installed on server or workstation operating systems, the VDA enables connections for desktops and apps. For Remote PC Access, install the VDA on the office PC.

Machine Creation Services (MCS) — A collection of services that work together to create virtual servers and desktops from a master image on demand, optimizing storage utilization and providing a pristine virtual machine to users every time they log on. Machine Creation Services is fully integrated and administrated in Citrix Studio.

Windows Server OS machines — VMs or physical machines based on Windows Server operating system used for delivering applications or hosted shared desktops to users.

Desktop OS machines — VMs or physical machines based on Windows Desktop operating system used for delivering personalized desktops to users, or applications from desktop operating systems.

Remote PC Access — User devices that are included on a whitelist, enabling users to access resources on their office PCs remotely, from any device running Citrix Receiver.

Additional components provide the following features:

Secure delivery — When users connect from outside the corporate firewall, this release can use Citrix NetScaler Gateway (formerly Access Gateway) technology to secure these connections with SSL. NetScaler Gateway or NetScaler VPX virtual appliance is an SSL VPN appliance that is deployed in the demilitarized zone (DMZ) to provide a single secure point of access through the corporate firewall.

WAN optimization — In deployments where virtual desktops are delivered to users at remote locations such as branch offices, Citrix CloudBridge (formerly Citrix Branch Repeater or WANScaler) technology can be employed to optimize performance. Repeaters accelerate performance across wide-area networks, so with Repeaters in the network, users in the branch office experience LAN-like performance over the WAN. CloudBridge can prioritize different parts of the user experience so that, for example, the user experience does not degrade in the branch location when a large file or print job is sent over the network. HDX WAN Optimization with CloudBridge provides tokenized compression and data deduplication, dramatically reducing bandwidth requirements and improving performance. For more information, see the Citrix CloudBridge documentation.
Features not in this release

Jan 03, 2014
The following features are not currently in or are no longer supported by XenDesktop 7 and related releases.

Features not in XenDesktop 7

- **Secure ICA encryption below 128-bit** — In previous releases, Secure ICA could encrypt client connections for basic, 40-bit, 56-bit, and 128-bit encryption. With this release, Secure ICA encryption is available only for 128-bit encryption.

- **Direct SSL connections** — In previous releases, administrators could configure SSL Relay support connections to Web Interface and between an SSL-enabled plug-in and each server. These types of connections are not supported in this release.

- **Legacy printing** — The following printing features are not supported in this release:
  - Backward compatibility for DOS clients and 16-bit printers, including legacy client printer name.
  - Support for printers connected to Windows 95 and Windows NT operating systems, including enhanced extended printer properties and Win32FavorRetainedSetting.
  - Ability to enable or disable auto-retained and auto-restored printers.
  - DefaultPrtmFlg, a registry setting for servers that is used to enable or disable auto-retained and auto-restored printers, which store in user profiles on the server.

- **Secure Gateway** — In previous releases, Secure Gateway was an option to provide secure connections between the server and user devices. NetScaler Gateway is the replacement option for securing external connections.

- **Shadowing users** — In previous releases, administrators set policies to control user-to-user shadowing. In this release, shadowing end-users is an integrated feature of the Director component, which uses Microsoft Remote Assistance to allow administrators to shadow and troubleshoot issues for delivered seamless apps and virtual desktops.

- **Web Interface** — StoreFront replaces Web Interface in XenDesktop 7 for aggregating apps and desktops to users.

- **Anonymous users** — In previous releases, administrators publishing an application could give access to the group called Anonymous, which allowed guest users permission to access applications without user authentication. In this release, no guest permissions are supported. Using Studio, administrators configure Delivery Groups and then allocate virtual desktops and applications to Delivery Groups.

- **Session pre-launch** — In previous releases, the session pre-launch feature could be used reduce application launch time during normal or high traffic periods. This feature is not available in this release.

- **Power and Capacity Management** — In previous releases, the Power and Capacity Management feature could be used to help reduce power consumption and manage server capacity. The Microsoft Configuration Manager is the replacement tool for this function.

- **Flash v1 Redirection** — Clients that do not support second generation Flash Redirection (including Receiver for Windows earlier than 3.0, Receiver for Linux earlier than 11.100, and Citrix Online Plug-in 12.1) will fall back to server-side rendering for legacy Flash Redirection features. VDAs included with this release support second generation Flash Redirection features.

- **Local Text Echo** — This feature was used to accelerate the display of input text on user devices on high latency connections. It is not included in XenDesktop 7 due to improvements to the graphics subsystem and HDX SuperCodec.

- **Virtual IP Loopback support** — In previous releases, this policy setting could allow each session to have its own virtual loopback address for communication. This policy is not available in this release. For a possible workaround on Windows Server 2012, see the Microsoft article: [http://social.technet.microsoft.com/wiki/contents/articles/15230.rds-ip-virtualization-in-windows-server-2012.aspx](http://social.technet.microsoft.com/wiki/contents/articles/15230.rds-ip-virtualization-in-windows-server-2012.aspx).

- **Smart Auditor** — In previous releases, Smart Auditor allowed you to record on-screen activity of a user’s session. This component is not available in this release.
• **Single Sign-on** — This feature, which provides password security, is not supported for Windows 8 and Windows Server 2012 environments. It is still supported for Windows 2008 R2 and Windows 7 environments, but is not included with this release. You can locate it on the Citrix download website: [http://citrix.com/downloads](http://citrix.com/downloads).

• **Oracle database support** — This release requires SQL database.

• **Health Monitoring and Recovery (HMR)** — In previous releases, HRM could run tests on the servers in a server farm to monitor their state and discover any health risks. In this release, Director offers a centralized view of system health by presenting monitoring and alerting for the entire infrastructure from within the Director console.

• **Custom ICA files** — Custom ICA files were used to enable direct connection from user devices (with the ICA file) to a specific machine. In this release, this feature is disabled by default, but can be enabled for normal usage using a local group or can be used in high-availability mode if the Controller becomes unavailable.

• **Management Pack for System Center Operations Manager (SCOM) 2007** — The management pack, which monitored the activity of farms using SCOM, does not support this release.

• **CNAME function** — The CNAME function was previously enabled by default. Deployments depending on CNAME records for FQDN rerouting and the use of NETBIOS names might fail. In this release, Delivery Controller auto-update is the replacement feature that dynamically updates the list of Delivery Controllers and automatically notifies VDAs when Controllers are added to and removed from the site. The Delivery Controller auto-update feature is enabled by default in Citrix policies, but can be disabled by creating a policy. Alternatively, you can re-enable the CNAME function in the registry to continue with your existing deployment and allow FQDN rerouting and the use of NETBIOS names. For more information, see CTX137960.

• **Quick Deploy wizard** — In previous releases of Studio, this option allowed a fast deployment of a fully installed XenDesktop deployment. The new simplified installation and configuration workflow in XenDesktop 7 eliminates the need for the Quick Deploy wizard option.

• **Native device drivers on Delivery Controllers** — To allow upgrades without restarting, no drivers are installed.

• **Remote PC Service configuration file and PowerShell script for automatic administration** — Remote PC is now integrated into Studio and the Delivery Controller.

• **Workflow Studio** — In previous releases, Workflow Studio was the graphical interface for workflow composition for XenDesktop. The feature is not supported with this release.

Features not in Citrix Licensing 11.11.1

• **Citrix Licensing Configuration Service** — This service displayed license information and enabled limited license server management in versions of Desktop Studio older than this new release of Studio. The old service is replaced in this release by Citrix Web Services for Licensing, which provides similar functionality. The enhanced tool also provides additional functionality. Citrix recommends upgrading to XenDesktop 7 or using the License Administration Console to manage licenses.

Features not in Receiver

• **COM Port Mapping** — COM Port Mapping allowed or prevented access to COM ports on the user device. COM Port Mapping was previously enabled by default. In this release, COM Port Mapping is disabled by default. For details, see [Configure COM Port and LPT Port Redirection settings using the registry](https://docs.citrix.com).

• **LPT Port Mapping** — LPT Port Mapping controls the access of legacy applications to LPT ports. LPT Port Mapping was previously enabled by default. In this release, LPT Port Mapping is disabled by default.

• **PCM Audio Codec** — Only HTML5 clients support the PCM Audio Codec in this release.

• **Support for Microsoft ActiveSync.**

• **Proxy Support for Older Versions** — This includes:
- Squid Proxy Server 3.1.14 (Ubuntu Linux Server 11.10).
Read this topic if you have used previous versions of XenDesktop. It describes new terminology used throughout the documentation and important changes and new features in XenDesktop.

XenDesktop now integrates Citrix XenApp and VDI desktop virtualization technologies into a unified architecture that enables a scalable, simple, efficient, and manageable solution for delivering Windows applications and desktops.

XenDesktop now enables mobile workstyles by allowing IT to deliver Windows desktops and applications as mobile services. Users can select applications from an easy-to-use “store” that is securely accessible from tablets, smartphones, PCs, Macs, and thin clients. XenDesktop delivers a native touch-optimized experience with HDX high-definition performance over mobile networks.

Learn new terms and names

**Citrix Director**—Monitoring console that provides detailed trend and diagnostic information on applications and desktops, and provides a detailed and intuitive overview of the new XenDesktop environments. Citrix Director replaces Desktop Director from earlier releases.

**Citrix Studio**—Management console that allows you to create and manage infrastructure and resources to deliver desktops and applications. Previously known as Desktop Studio.

**Delivery Controller**—A server-side component that is responsible for distributing desktops and applications to users, managing user access through policies, managing desktops, and managing reboot cycles for servers. You may also see the Delivery Controller referred to as the Controller. Previously known as controller and Desktop Delivery Controller.

**Delivery Group**—The container for a machine or machines used to deliver applications and desktops to a specific group of users. Previously known as desktop groups or assignments.

**Desktop OS machines**—Virtual or physical machines based on Windows Desktop operating system used for delivering personalized desktops or applications from desktop operating systems to users.

**Host**—The hypervisor that hosts the master images and virtual machines.

**Machine Catalog**—A collection of machines. The machine catalog can be used to create applications and desktop for users.

**Machine Creation Services**—A service that runs on Delivery Controller and uses a master image to create a server or desktop machine(s) with a unique identity on the network and in Active Directory.

**Master Image**—An installed instance of a server or desktop operating system, including all the applications and the Virtual Delivery Agent (optional) installed. The master image can be duplicated using a provisioning technology to create machines that host applications and desktops to end users.

**NetScaler Gateway**—Provides secure access to remote users. Formerly called Access Gateway.

**Provisioning Services**—A service that allows you to create virtual or physical instances of desktop or server machines.

**Site**—The core XenDesktop environment consisting of the Delivery Controller and Database.
Virtual Delivery Agent — The software agent that is installed on the virtual or physical machine that provides the virtual desktop or application to the user. The acronym VDA is frequently used. Previously known as Virtual Desktop Agent.

Windows Server OS machines — Virtual or physical machines based on Windows Server operating system used for delivering applications or hosted shared desktops to users.

Key changes and additions

XenDesktop contains powerful new administration features to automate common tasks while integrating the configuration of profile management, StoreFront, publishing, and storage optimization. Among many new features, XenDesktop introduces two consoles for service design (Citrix Studio) and performance monitoring (Citrix Director) and a unified architecture that enables the most scalable solution for delivering Windows applications and desktops as cloud services.

XenDesktop offers users complete virtual desktops or just applications. By using Citrix Receiver with XenDesktop, users have a device-native experience on devices including Windows, Mac, Linux, iOS, Android, Chrome OS, Blackberry.

- **Studio** — The management console that enables you to configure and manage your deployment, eliminating the need for separate management consoles for managing delivery of applications and desktops. Studio provides various wizards to guide you through the process of setting up your environment, creating your workloads to host applications and desktops, and assigning applications and desktops to users. Studio empowers IT service designers with simple workflows and automated configuration checking to eliminate as much as 80% of the time and effort needed to deploy new services.
- **Director** — Monitor and troubleshoot deployments using Director's redesigned user interface and integrated EdgeSight functionality:
  - A new help desk view offers an improved troubleshooting experience for help desk administrators, allowing user, machine, and application issues to be resolved quickly.
  - Full administrators have access to other views, including a newly designed Dashboard to provide a graphical summary of your deployment in a central location, and a Trends page to provide improved, in-depth, graphical monitoring and troubleshooting of the entire deployment with various time ranges.
  - HDX Insight is the integration of EdgeSight Network Inspector and EdgeSight Performance management with Director to provide real-time assessment and historical trending and advanced analytics of ICA traffic in your deployment.
- **Mobile optimizations** — Optimizations translate multitouch gestures, finger swipes, and inertia into smooth navigation and scrolling commands that are relayed to the hosted Windows applications in real-time.
- **HDX enhancements** — New technology overcomes the challenges of mobile networks, by cutting bandwidth consumption, while increasing high definition video frame rates and dynamically conquering the variations in latency and dropped packets that are present in public broadband networks.
- **Desktop Composition Redirection** — Extend the Aero Redirection feature introduced in XenDesktop 5.5 to provide users with a more fluid Windows 7 Aero or Windows 8 desktop experience by leveraging the graphics processing unit (GPU) or integrated graphics processor (IGP) on supported Windows client devices for local DirectX graphics rendering. This feature delivers rich multimedia applications to user devices, while maintaining high scalability on the server.
- **Windows Media client-side content fetching** — Enable a user device to stream multimedia files directly from the source provider on the Internet or Intranet to improve network utilization and server scalability.
- **Multicast support** — Reduce bandwidth consumption when streaming live video to branch offices. Multicast support enables a single Windows Media source transmission to support multiple users.
• **Real-time multimedia transcoding** — Enable audio and video streaming to mobile devices, improving the way that Windows Media content is delivered. Host-based transcoding provides a seamless user experience, even in extreme network conditions. To improve server scalability, if the Virtual Delivery Agent has a supported graphics processing unit (GPU) for hardware acceleration, transcoding is done in the GPU.

• **User Datagram Protocol (UDP) audio for Server OS machines** — Extend support for audio delivery over UDP/RTP to Server OS machines. This feature delivers superior audio quality for real-time applications like video conferencing and streaming media, even in environments when there is packet loss or congestion.

• **Webcam video compression** — Reduce bandwidth consumption to improve performance when using supported video conferencing applications.

• **HDX 3D Pro** — Deliver applications with graphics processing units (GPUs) for hardware acceleration to the desktop. This includes 3D professional graphics applications based on OpenGL and DirectX.

• **Server-rendered Rich Graphics and Video** — Deliver rich graphics (including Windows 8 or Windows Aero) to virtual desktop users on any client, platform, and bandwidth condition by rendering the content on the server. When users have the latest Citrix Receiver, they will also notice improved performance for server-rendered video.

• **Improved Flash Redirection** — Determine when to redirect Adobe Flash content to the user device for local rendering based on the user's network type and environment. Flash Redirection helps reduce server and network load, resulting in greater scalability while ensuring a high-definition user experience.

### Machine Catalog Desktops and Applications

• **Desktops** — Deliver managed desktops to multiple, simultaneously connected remote users. Server OS machine and Desktop OS machine desktops provide features such as SmoothRoaming, session reliability, license consumption, Receiver and Delivery Services authentication, and Desktop Viewer. This release introduces Windows Server OS machine catalogs and desktops as a cost-effective virtualization delivery solution because they provide a significant number of remote users per server and storage resource. This release also provides Remote PC desktops that let users access resources on their office PCs remotely, from any device running Citrix Receiver.

• **Applications** — Integrate XenApp publishing capabilities within a deployment. Use this feature to deliver shared hosted applications to multiple, simultaneously-connected remote users.

• **Redesigned Citrix StoreFront** — Unifies application and desktop access through a simple, consistent user experience, including seamless transitions between devices, which pick up where the user left off.

• **StoreFront replaces Web Interface** — Create centralized enterprise stores to deliver desktops, applications, and other resources to users on any device. The feature provides authentication and resource delivery services for Citrix Receiver.

• **Desktop Appliance sites** — Access Desktop Appliance sites through a website on the StoreFront store. The site is created automatically when a store is created. If, for example, the store has a path of path/Citrix/Store, the Desktop Appliance site path is path/Citrix/StoreDesktopAppliance. Users can restart their virtual machine-hosted desktops on Desktop Appliance sites.

• **Database as a service** — Write user subscription data for each store to the local disk on the StoreFront server by using the new subscription store service. The data is then propagated across the server group.

• **Configuration Logging** — Capture Site configuration changes and administrative activities to a configuration logging database. You can view the log in Studio using a variety of filters and generate HTML and CSV reports.

• **Delegated Administration** — Group objects into administrative scopes. This feature provides an enterprise-class administration model with role-based access control, custom roles with configurable permissions, and fine-grained, object-based control.

• **New installer** — A single, streamlined installer to guide you through installing the core components (Delivery Controller, Studio, Director, StoreFront, and License Server) and VDAs.

• **AppDNA** — Included as part of some XenDesktop editions to evaluate and solve compatibility issues for deploying desktop and web applications, or for operating system upgrades.
• **Microsoft Application Virtualization (App-V) integration** — Lets you deploy, update, and support App-V applications. Using Studio, you configure access to App-V Management and Publishing servers and then deliver the App-V applications to users through Delivery Groups along with locally installed applications.

• **Local App Access** — Enables the seamless integration of users' locally-installed applications and hosted applications within a hosted desktop environment.

Components in this release
System requirements for XenDesktop 7.1

Mar 12, 2015
This topic lists the supported systems for the Delivery Controller, Citrix Studio, Citrix Director, and Virtual Delivery Agents (VDAs), at the time this XenDesktop version released.

System requirements for other features and components, such as StoreFront, host systems, HDX, receivers and plug-ins, Desktop Lock, and Provisioning Services are described in their respective documentation.

Unless otherwise noted, the installer deploys component prerequisites automatically. This includes .NET 3.5 SP1, which is required when installing components on Windows 7 or Windows Server 2008 R2 systems.

The installation media contains several third-party components. Before using the Citrix software, check for security updates from the third party, and install them.

If you install all the core components (Controller, Studio, Director, StoreFront, and Licensing) on a single server, you need a minimum of 3 GB of RAM to evaluate the product; more is recommended when running an environment for users. Performance will vary depending on your exact configuration, including the number of users, applications, desktops, and other factors.

Delivery Controller

Supported operating systems:
- Windows Server 2012 R2, Standard and Datacenter Editions.
- Windows Server 2008 R2 SP1, Standard, Enterprise, and Datacenter Editions.

Requirements:
- Disk space: 100 MB.
- Microsoft .NET Framework 3.5.1 (required on Windows Server 2008 R2 only).
- Microsoft .NET 4.0.
- Windows PowerShell 2.0 (included with Windows Server 2008 R2) or 3.0 (included with Windows Server 2012 R2 and Windows Server 2012).
- Visual C++ 2005, 2008 SP1, and 2010 Redistributable packages. The installer deploys these automatically. They are also available on the Citrix installation media in the Support folder.

Database

Supported Microsoft SQL Server versions for the Site Configuration Database (which initially includes the Configuration Logging Database and the Monitoring Database):
- SQL Server 2012 SP1, Express, Standard, and Enterprise Editions. By default, SQL Server 2012 Express is installed when installing the Controller.
- SQL Server 2008 R2 SP2, Express, Standard, Enterprise, and Datacenter Editions.

The following database features are supported (except on SQL Server Express Edition, which supports only standalone mode).
- SQL Server Clustered Instances.
- SQL Server Mirroring.
- SQL Server 2012 AlwaysOn Availability Groups.
Windows authentication is required for connections between the Controller and the SQL Server database.

For information about the latest supported database versions, see CTX114501.

Studio

Supported operating systems:
- Windows 8.1, Professional and Enterprise Editions.
- Windows 8, Professional and Enterprise Editions.
- Windows 7 Professional, Enterprise, and Ultimate Editions.
- Windows Server 2012 R2, Standard and Datacenter Editions.
- Windows Server 2008 R2 SP1, Standard, Enterprise, and Datacenter Editions.

Requirements:
- Disk space: 75 MB.
- Microsoft .NET Framework 3.5 SP1 (required on Windows Server 2008 R2 only).
- Microsoft Management Console 3.0 (included with all supported operating systems).

Director

Supported operating systems:
- Windows Server 2012 R2, Standard and Datacenter Editions.
- Windows Server 2008 R2 SP1, Standard, Enterprise, and Datacenter Editions.

Requirements:
- Disk space: 50 MB.
- Microsoft .NET Framework 4.0. The installer deploys this automatically. It is also available on the Citrix installation media in the Support\DotNet4 folder.
- Microsoft Internet Information Services (IIS) 7.0 and ASP.NET 2.0. If these are not already installed, you are prompted for the Windows Server installation media, then they are installed for you.
- Supported browsers for viewing Director:
  - Internet Explorer 11, 10, and 9 (compatibility mode in Internet Explorer 10 is not supported)
  - For Internet Explorer 10, compatibility mode is not supported. You must use the recommended browser settings; otherwise, you cannot access Director. When you install Internet Explorer 10, accept the default to use the recommended security and compatibility settings. If you already installed the browser and chose not to use the recommended settings, go to Tools > Internet Options > Advanced > Reset and follow the instructions.
    - Firefox
    - Chrome

Virtual Delivery Agent (VDA) for Windows Desktop OS

Supported operating systems:
- Windows 8.1, Professional and Enterprise Editions.
- Windows 8, Professional and Enterprise Editions.
- Windows 7 SP1, Professional, Enterprise, and Ultimate Editions.
The installer automatically deploys the following requirements, which are also available on the Citrix installation media in the Support folders:

- (For Windows 7 systems only.) Microsoft .NET Framework 3.5.1 - if using the command-line interface, you must install this manually before installing the VDA.
- Microsoft .NET Framework 4.0.
- Microsoft Visual C++ 2010 Runtime (32-bit or 64-bit, depending on platform).

Remote PC Access uses this VDA, which you install on physical office PCs.

Several multimedia acceleration features (such as HDX MediaStream Windows Media Redirection) require that Microsoft Media Foundation be installed on the machine where you install the VDA. If the machine does not have Media Foundation installed, the multimedia acceleration features will not be installed and will not work. Do not remove Media Foundation from the machine after installing the Citrix software; otherwise, users will not be able to log on to the machine. On most Windows 8.1, Windows 8, and Windows 7 editions, Media Foundation support is already installed and cannot be removed. However, N editions do not include certain media-related technologies; you can obtain that software from Microsoft or a third party.

You cannot install a XenDesktop 7.1 VDA on a machine running Windows XP or Windows Vista; however, you can install an earlier Virtual Desktop Agent version on those operating systems, if needed. For more information, see the instructions for installing an earlier VDA version on those operating systems. The Remote PC Access version in this release is not supported on Windows Vista operating systems.

Virtual Delivery Agent (VDA) for Windows Server OS

Supported operating systems:
- Windows Server 2012 R2, Standard and Datacenter Editions.
- Windows Server 2008 R2 SP1, Standard, Enterprise, and Datacenter Editions.

The installer automatically deploys the following requirements, which are also available on the Citrix installation media in the Support folders:

- (For Windows 2008 R2 systems only.) Microsoft .NET Framework 3.5.1 - if using the command-line interface, you must install this manually before installing the VDA.
- Microsoft .NET Framework 4.0.

The installer automatically installs and enables Remote Desktop Services role services, if they are not already installed and enabled.

Several multimedia acceleration features (such as HDX MediaStream Windows Media Redirection) require that the Microsoft Media Foundation be installed on the machine where you install the VDA. If the machine does not have Media Foundation installed, the multimedia acceleration features will not be installed and will not work. Do not remove Media Foundation from the machine after installing the Citrix software; otherwise, users will not be able to log on to the machine. On most Windows Server 2012 R2, Windows Server 2012, and Windows Server 2008 R2 editions, the Media Foundation feature is installed through the Server Manager (for Windows Server 2012 R2 and Windows Server 2012: ServerMediaFoundation, for Windows Server 2008 R2: DesktopExperience). However, N editions do not include certain media-related technologies; you can obtain that software from Microsoft or a third party.

The Print Spooler Service is enabled by default on the Windows server. If you disable this service, you cannot successfully install a VDA for Windows Server OS. Therefore, ensure that this service is enabled before installing a VDA.
Host

Supported platforms:
- XenServer.
  - XenServer 6.2
  - XenServer 6.1
  - XenServer 6.0.2
- VMware vSphere. No support is provided for vSphere vCenter Linked Mode operation.
  - VMware vSphere 5.1 Update 3
  - VMware vSphere 5.1 Update 2
  - VMware vSphere 5.1 Update 1
  - VMware vSphere 5.0 Update 3
  - VMware vSphere 5.0 Update 2
- System Center Virtual Machine Manager - Includes any version of Hyper-V that can register with the supported System Center Virtual Machine Manager versions.
  - System Center Virtual Machine Manager 2012 R2
  - System Center Virtual Machine Manager 2012 SP1
  - System Center Virtual Machine Manager 2012

See the platform documentation for system requirements. See CTX131239 for updated hypervisor support information.

The following Host and storage technology combinations are supported for Machine Creation Services and runtime Active Directory account injection into VMs. Combinations marked with an asterisk (*) are recommended.

<table>
<thead>
<tr>
<th>Host</th>
<th>Local Disks</th>
<th>NFS</th>
<th>Block Storage</th>
<th>Storage Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>XenServer</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>VMware ESX</td>
<td>Yes (no vMotion or dynamic placement)</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Hyper-V</td>
<td>Yes</td>
<td>No</td>
<td>Yes * (requires Cluster Shared Volumes)</td>
<td>No</td>
</tr>
</tbody>
</table>

Other

- StoreFront requires 2 GB of memory. See the StoreFront documentation for full system requirements. StoreFront 2.0 is the minimum version supported with this version of XenDesktop.
- Only Citrix License Server for Windows is supported. The minimum supported version is 11.11. The Citrix License Server requires 40 MB of disk space. See the licensing documentation for full system requirements.
- Universal Print Server - Supported operating systems (the Controller includes the Universal Print Server functionality; you need only install the Universal Print Server on your print servers):
  - Windows Server 2008 R2 SP1
  - Windows Server 2008 32-bit
- The Microsoft Group Policy Management Console (GPMC) is required if you store Citrix policy information in Active Directory rather than the Site Configuration Database. For more information, see the Microsoft documentation.
- You can install the Receiver for Windows when installing a VDA provided on the XenDesktop installation media. For
system requirements information on other platforms, see the Receiver for Windows documentation.

- The Receiver for Linux and the Receiver for Mac are provided on the XenDesktop installation media. See their documentation for system requirements.
- When using Provisioning Services with this version of XenDesktop, the minimum supported Provisioning Services version is 7.0.
- When using Access Gateway versions earlier than 10.0 with this version of XenDesktop, Windows 8.1 and Windows 8 clients are not supported.
How do I...

Mar 28, 2015

- Upload error reports to Citrix
- Track system changes and use the information in Director for root cause analysis
- Delegate administration
- Improve the audio experience for Windows 7 and 8
- Improve the visual experience for Windows 7 and 8
- Deliver a 3D graphics experience on Windows 7 and 8
- Offer the best experience for mobile devices on 3G and 4G networks
- Enable smart card authentication
- Allow users to connect to their office PCs from home
- Allow users to install their own applications on a virtual desktop
- Deliver a Windows 7 look and feel desktop from a server hosted desktop
- Deliver applications from Windows Server Operating Systems
- Stream applications from App-V
- Deliver 32-bit or 16-bit applications
- Allow users to access locally installed applications from a short cut on their virtual desktops
- Improve the user experience for mobile devices
- Create a list of changes over time
- Connect users to their local physical PCs and laptops using HDX for a remote access solution
- Offer improved graphics experience for a large number of users or users who use graphic intensive applications
- How do I orchestrate and automate desktop and application delivery
Get started

Mar 28, 2015

Current XenDesktop users upgrading to XenDesktop 7.1

Some of the things you want to do first include:
- Identify new terms and concepts, terms that have changed with this version, and components included in this version. See XenDesktop concepts.
- Learn how to publish desktops and applications. See the following sections:
  - Set up machine catalogs
  - Delivery Groups
  - Hosted applications
- Learn how to upgrade from an earlier XenDesktop installation. See Upgrade XenDesktop 5.

New XenDesktop users

You may be a Citrix XenApp administrator moving to XenDesktop or you may never have used Citrix products before. Some of the things you want to do first include:
- Learn XenDesktop terminology. See XenDesktop concepts.
- Learn about XenDesktop concepts, components, and architecture. See Important information for XenApp administrators.
- Learn about master images, Machine Catalogs, and Delivery Groups. See the following sections:
  - Set up machine catalogs
  - Delivery Groups
- Learn how to publish desktops and applications. See Create a Delivery Group application.
**XenDesktop concepts**

Oct 22, 2013

Concepts of publishing

**Master Image**

A master image is an image used by the provisioning technology to create virtual machines (VMs) for your users. Depending on the provisioning technology used, the master image can also be used to create a machine to host applications and desktops. The master image, created and stored on your hypervisor, contains the operating system and common applications and settings you are providing to your users, such as:

- Anti-virus software.
- Citrix plug-ins.
- Other default programs.

With a master image, all users start with desktops that are created from the master image. Depending on the machine catalog type, any user customization and system updates performed on the desktop are either persisted or discarded when users log off.

Applications and desktops on the master image are securely managed, hosted, and run on machines within your datacenter, providing a more cost effective application delivery solution.

**Provisioning methods**

This release supports these provisioning methods:

- **Machine Creation Services (MCS)**—This method uses a master image within your environment to manage virtual machines, enabling you to manage and update target devices through one master image. Machine Creation Services is fully integrated and administrated in Citrix Studio.

- **Provisioning Services (PVS)**—This method allows computers to be provisioned and re-provisioned in real-time from a single shared-disk image. Provisioning Services manages target devices as a device collection. The desktop and applications are delivered from a Provisioning Services vDisk imaged from a master target device, and enables you to leverage the processing power of physical hardware or virtual machines. Provisioning Services is managed through its own console.

- **Existing images**—This method manages and delivers desktops and applications that you have already migrated to virtual machines in the data center. You must manage target devices on an individual basis or collectively using third-party electronic software distribution (ESD) tools.

Select your provisioning method when creating a Machine Catalog.

**Machine catalogs**

A machine catalog is a collection of virtual machines and physical machines managed as a single entity. Machine catalogs specify:

- The virtual or physical machines available to host applications or desktops
- The Active Directory computer accounts assigned to those virtual machines or computers
In some cases, the master image that is copied to create the virtual machines

There are two catalogs:

- **Windows Server OS Machine catalog**—Virtual or physical machines based on Windows Server operating system used for delivering applications or hosted shared desktops to users.
- **Desktop OS Machine catalog**—Virtual or physical machines based on Windows Desktop operating system used for delivering personalized desktops to users, or applications from desktop operating systems.

### Delivery Groups

Machines within machine catalogs are organized into Delivery Groups. Delivery Groups deliver the same set of applications or desktops to groups of users.

In a Delivery Group, you can:

- Assign specific teams, departments, or types of users to desktops or applications.
- Use machines from multiple catalogs.
- Allocate a user to multiple machines.
- Allocate multiple users to one machine.

Server OS machines support:

- Desktop and applications Delivery Groups that host both desktops and applications.
- Application Delivery Groups that host only applications.

Desktop OS machines support:

- The same as Server OS machines.
Important information for XenApp administrators

Mar 28, 2015

This release unifies hosted applications and desktops (XenApp) with personalized desktops (XenDesktop) within a single architecture and management experience. The capabilities previously available within XenApp are now delivered within the XenDesktop infrastructure and components. If you have prior experience managing XenApp farms, this topic helps you understand the following:

- Key concepts of the unified infrastructure
- Differences in terminology
- The components and what they do
- Application delivery using the new infrastructure
- Monitoring and managing multiple versions of XenApp and XenDesktop

To review some of the past XenApp features that are deprecated in this release, see Features not in this release.

For system requirements of the components in this release, see System requirements for XenDesktop 7.1.

Important infrastructure concepts for XenApp administrators

The following terms and concepts are important for you to understand so that you can effectively publish applications and desktops from the unified infrastructure. When you publish applications, you are creating or adding applications. These components are:

- A machine catalog is a collection of virtual machines (VMs) and physical machines managed as a single entity. Machine catalogs specify the following:
  - VMs or physical computers available to host applications or desktops
  - The Active Directory computer accounts assigned to those VMs or computers
  - In some cases, the master image that is copied to create the VMs
- A Delivery Group is a collection of servers that specify who can use a set of applications. A single Delivery Group can contain applications from a number of machine catalogs rather than a single hypervisor pool. Also, a single Delivery Group can be published to users so that a single user can access multiple applications in the group.

Key differences between XenApp and XenDesktop

In moving from a traditional XenApp environment to the unified environment, you will find the following key differences:

- **Citrix Studio instead of Delivery Services Console**— In this release, you use Studio to configure your environments and provide users with access to applications and desktops. For example, instead of using folders and Worker Groups to organize applications, servers, and other resources, in Studio you organize those resources using a combination of machine catalogs, tags, Delivery Groups, and Delegated Administrators.

- **Delegated Administration** — In XenApp, you can create custom administrators and assign them permissions based on folders and objects. In XenDesktop, you can create custom administrators whose permissions are based on role and scope pairs. A role represents a job function and has defined permissions associated with it. A scope represents a collection of objects. You use scopes to group objects in a way that is relevant to your organization (for example, the set of Delivery Groups used by the Sales team). This release also offers several built-in administrator roles (other than the full administrator role), such as help desk, applications, hosting, and catalog. Each of these built-in roles includes specific
management permissions. For more information, see Delegated Administration.

- **No IMA data store** — This release does not use the IMA data store as the central database in which to store configuration information. Instead, it uses a Microsoft SQL Server database as the data store for both configuration and session information. This means:
  - Database requirements are different. Microsoft Access and Oracle are no longer supported databases.
  - Remote Desktop Services Client Access Licenses (RDS CALs) are no longer needed on the servers where Controllers are running. You still need RDS CALs on the servers that are hosting and delivering your applications and desktops.
  - There is no dedicated zone master. In XenApp, there is a zone master or data collector responsible for user connection requests and communication with hypervisors. In this release, this function is distributed evenly across all Controllers in the site.
  - If you require high availability or disaster recovery for Microsoft SQL Server you can configure clustering or mirroring, or deploy the database as a virtual machine and use your hypervisor's high availability features instead. For more information, see Ensure database fault tolerance.

- **FlexCast Management Architecture (FMA)** — FMA requires that you must be in a domain to deploy a site. For example, to install the servers, your account must have local administrator privileges and be a domain user in the Active Directory.

- **Sites instead of Farms** — In this release, the XenApp "farms" are known as "sites." Sites should be within one data center.

- **Citrix Director monitors the environment** — Director is a separate monitoring tool that is installed by default as a website on the Delivery Controller. From this console, administrators (depending on their Delegated Administrator permissions) can monitor the environment, shadow user devices, and troubleshoot IT issues for users and sites. For example, help desk administrators can work only with individual users on specified sites, while full administrators can monitor the entire deployment and resolve system-wide IT issues. For more information on Director and how it enables IT support, see Monitor environments with Director.

- **No Shadow taskbar** — To view and interact with other users' sessions remotely, you use the shadow feature launched from Director console, which uses Microsoft Remote Assistance to connect to user machines. Remote Assistance is installed by default on the VDA; if you clear the check box during installation, you cannot shadow the user remotely.

Components in this release

If you already have experience with a XenDesktop or XenApp environment, it will be helpful to you to identify the components introduced in this release and learn how they work and communicate with each other. Under the new architecture, XenDesktop and XenApp are unified, including management and delivery components, to give administrators a unified management experience.

This figure shows the key components in a typical deployment.
The core components of this release are:

**Director** — Director is a web-based tool that enables IT support and help desk teams to monitor an environment, troubleshoot issues before they become system-critical, and perform support tasks for end users. You can also view and interact with a user's sessions using Microsoft Remote Assistance.

**Receiver** — Installed on user devices, Citrix Receiver provides users with quick, secure, self-service access to documents, applications, and desktops from any of the user's devices including smartphones, tablets, and PCs. Receiver provides on-demand access to Windows, Web, and Software as a Service (SaaS) applications.

**StoreFront** — StoreFront authenticates users to sites hosting resources and manages stores of desktops and applications that users access.

**Studio** — Studio is the management console that enables you to configure and manage your deployment, eliminating the need for separate management consoles for managing delivery of applications and desktops. Studio provides various wizards to guide you through the process of setting up your environment, creating your workloads to host applications and desktops, and assigning applications and desktops to users.

**License server** — License server manages your product licenses. You must create at least one license server to store and manage your license files.

**Delivery Controller** — Installed on servers in the data center, the Delivery Controller consists of services that communicate with the hypervisor to distribute applications and desktops, authenticate and manage user access, and broker connections between users and their virtual desktops and applications. The Controller manages the state of the desktops, starting and stopping them based on demand and administrative configuration. In some editions, the Controller allows you to install Profile management to manage user personalization settings in virtualized or physical Windows environments. Each site has
one or more Delivery Controllers.

**XenServer** — XenServer is an enterprise-class virtual machine infrastructure solution that creates the foundation for delivering virtual desktops and offers advanced management features. Multiple VMs can run on XenServer, which takes advantage of the advanced virtualization features of the latest virtualization-enabled processors from Intel and AMD. For more information about XenServer, see the XenServer documentation in eDocs.

**Virtual Delivery Agent (VDA)** — Installed on server or workstation operating systems, the VDA enables connections for desktops and apps. For Remote PC Access, install the VDA on the office PC.

**Machine Creation Services (MCS)** — A collection of services that work together to create virtual servers and desktops from a master image on demand, optimizing storage utilization and providing a pristine virtual machine to users every time they log on. Machine Creation Services is fully integrated and administered in Citrix Studio.

**Windows Server OS machines** — VMs or physical machines based on Windows Server operating system used for delivering applications or hosted shared desktops to users.

**Desktop OS machines** — VMs or physical machines based on Windows Desktop operating system used for delivering personalized desktops to users, or applications from desktop operating systems.

**Remote PC Access** — User devices that are included on a whitelist, enabling users to access resources on their office PCs remotely, from any device running Citrix Receiver.

Additional components provide the following features:

**Secure delivery** — When users connect from outside the corporate firewall, this release can use Citrix NetScaler Gateway (formerly Access Gateway) technology to secure these connections with SSL. NetScaler Gateway or NetScaler VPX virtual appliance is an SSL VPN appliance that is deployed in the demilitarized zone (DMZ) to provide a single secure point of access through the corporate firewall.

**WAN optimization** — In deployments where virtual desktops are delivered to users at remote locations such as branch offices, Citrix CloudBridge (formerly Citrix Branch Repeater or WANScaler) technology can be employed to optimize performance. Repeaters accelerate performance across wide-area networks, so with Repeaters in the network, users in the branch office experience LAN-like performance over the WAN. CloudBridge can prioritize different parts of the user experience so that, for example, the user experience does not degrade in the branch location when a large file or print job is sent over the network. HDX WAN Optimization with CloudBridge provides tokenized compression and data deduplication, dramatically reducing bandwidth requirements and improving performance. For more information, see the Citrix CloudBridge documentation.

**Application delivery in this release**

In XenApp, you use the Publish Application wizard to prepare applications and deliver them to individuals or groups of users. In XenDesktop, you use the Studio component to create and add applications to make them available to users who are included in a Delivery Group. Using Studio, you first configure a site, create and specify machine catalogs, and then create Delivery Groups within those machine catalogs. Delivery Groups are then used to determine which users have access to the applications you deliver. Refer to the Deliver section for this release.

Once you have created Delivery Groups, you can then create an application to specify which Delivery Groups will provide access to that specific application. In Studio, you can view which applications are added to a selected Delivery Group, as
well as enable, disable, or modify the applications. For a detailed explanation of how you configure Delivery Groups to make applications available to users, see Delivery Groups.

These methods are available for delivering applications, and you can choose the method that is most appropriate for your specific environment:

- **Hosted applications and desktops** — This server-hosted applications feature integrates XenApp functionality into a XenDesktop environment to deliver applications virtually to remote user devices in high definition. For details, review the Plan section for this release, and in particular, see Plan for hosting desktops and applications.

- **Local App Access** — This client-hosted applications feature enables the integration of users' locally-installed applications and hosted applications within a hosted desktop environment. This feature allows users to access all of the applications they need in one place. Using Local App Access, you can publish shortcuts to locally-installed applications on a virtual desktop. When a user connects to a virtual desktop, shortcuts to locally-installed applications are displayed on the virtual desktop and in the Start menu. When the user launches a local application in the virtual desktop session, the application window appears in the desktop session window even though it is actually running on the user's computer.

- **Streamed applications** — This release supports App-V 5.0 as the preferred technology to stream applications to user devices. If you use the application streaming feature in XenApp deployments, including the Streaming Profiler and Offline Plug-in, you can continue to use that process in those deployments.

**Manage multiple versions of XenApp and XenDesktop**

There is no XenApp to XenDesktop 7 upgrade. Citrix will support customers in their migration from XenApp 6.5 to XenDesktop in a future release and plans to release tools and or scripts to assist in this transition.

The Studio management and Director can monitor and manage only XenDesktop 7 sites. The monitoring and management tools do not support past versions of XenDesktop or XenApp.

For example, XenDesktop 7 Director requires a XenDesktop 7 Delivery Controller. Director 7 can monitor XenDesktop 5.x VDAs; however, some data, including logon duration, will not be available with the XenDesktop 5.x VDAs.

Citrix recommends that if you continue running deployments of past versions of XenApp or XenDesktop, you run them in parallel with the XenDesktop 7 Site and continue running the management consoles with each release for that site.

For example, in a mixed environment, to continue using Desktop Director 2.1 to monitor XenApp 6.5, make sure that Desktop Director 2.1 is installed on a separate server from Director 7.

Use StoreFront to aggregate applications and desktops from the different versions of XenApp and XenDesktop. For details, see the StoreFront section.
Upgrade XenDesktop 7

Mar 28, 2015

Review the following information before upgrading XenDesktop 7:

- **Existing Sites:** You must use the procedure known as an in-place upgrade to upgrade to XenDesktop 7.1 or 7.5.
- **VDAs:** You cannot upgrade Virtual Desktop Agents running on Windows XP or Windows Vista to version 7.1 Virtual Delivery Agents. You must upgrade these VDAs to the Windows XP or Windows Vista version provided by the installer, or upgrade them to Version 5.6 Feature Pack 1.
- **Microsoft System Center Virtual Machine Manager (SCVMM):** XenDesktop 7.1 and 7.5 support SCVMM 2012, SCVMM 2012 SP1, and SCVMM 2012 R2.

When you run the installer AutoSelect, the wizard checks whether certain Site components (such as the Delivery Controllers, Director, and VDAs), need to be upgraded. If you opt not to upgrade some components during this process, when you run Studio, it performs a component check and notifies you when components need to be upgraded. You cannot proceed to manage your Site until you upgrade these components.

**Important:** Back up your Databases as described in How to backup and Restore your XenDesktop Database before performing any upgrade procedures.

**Upgrade Site with components deployed on different machines**

The following figure shows the high-level processes involved when upgrading a XenDesktop 7 to XenDesktop 7.x in which Studio components are deployed on different machines.
Perform the upgrade procedures in the following order.

Check the licenses
Make sure that your Subscription Advantage date for licenses is no earlier than 2013.0522.

Upgrade StoreFront
If StoreFront is deployed on a separate machine, follow the steps described in Upgrade StoreFront.

Upgrade Director
If Director is deployed on a separate machine, follow the steps described in 
— Upgrade core components
.

Upgrade Provisioning Services
If you are using Provisioning Services, follow the procedures described in Upgrading Provisioning Services.

Upgrade the Virtual Delivery Agent
For Remote PC Access deployments, Citrix recommends that you upgrade a VDA in a remote PC using a command-line method. For detailed information, see Install using the command line.

1. Log on to the server using a local administrator account and run the installer by inserting the media or mounting the ISO drive for this release, and double-clicking AutoSelect.
2. On the Welcome page, click Start. The wizard detects what components need to be upgraded and displays the Upgrade options page, activating the components you can upgrade.
3. On the Upgrade options page, select:
   - Virtual Delivery Agent for Windows Desktop OS for Desktop OS, and earlier XenDesktop versions
   - Virtual Delivery Agent for Windows Server OS for Server OS
4. On the Firewall page review the default ports and configure firewall rules.
5. On the Summary review the prerequisites to be installed and the components to be upgraded then click Upgrade.
6. On the Finish Upgrade page one of the following messages appears upon completion:
   - Success — Upgrade successful appears when the upgrade completes without errors.
   - Failed — The Upgrade failed appears with a list of failed components. Click Why did this fail to review what you must do to fix the problem. Other components that installed successfully are retained; you do not need to reinstall them.
7. Click Finish to complete the upgrade.

Upgrade Delivery Controllers

Citrix recommends that you upgrade Delivery Controllers as follows:
1. Upgrade half of your Site's Delivery Controllers.
2. Use the upgraded version of Studio to perform the Site upgrade as described in — Upgrade core components.
3. Upgrade your remaining Delivery Controllers.

Upgrade Databases using Studio

Use Studio to upgrade the Database.

Automatically upgrade Databases

1. Start Studio. The wizard detects what components need to be upgraded and displays the Mandatory upgrade page.
2. Select Start the Site upgrade automatically.
3. At the prompt, select I am ready to upgrade.
   - The wizard displays the upgrade progress. After the upgrade completes, the wizard performs tests. This takes several minutes.
4. At the Site Upgrade Complete window, you can optionally view a data migration report and then click Finish.
5. At the Upgrade successful page, click Finish upgrade and return to the Site overview.

Manually upgrade Databases

To minimize Site down time when performing a manual upgrade, it is important that the XenDesktop Administrator closely coordinates with the Database Administrator. This process requires that you run a script that temporarily disables XenDesktop Services while the manual upgrade scripts are run by the Database Administrator using a preferred tool (for example, SQL Server Management Studio). Ideally, immediately after these scripts complete, the XenDesktop Administrator
should enable XenDesktop Services and complete the upgrade using Studio.

Manual upgrade requires:

- Backing up the databases
- Running the generated scripts in the following order:

1. DisableServices.ps1 — PowerShell script to be run by the XenDesktop administrator on an XenDesktop Controller.
2. UpgradeSiteDatabase.sql — SQL script run where the Site databases resides
3. UpgradeMonitorDatabase.sql — SQL script run where the Monitor databases resides.
4. UpgradeLoggingDatabase.sql — SQL script run where the Logging database resides.
   Note: You should only run this script if the Logging database changes. For example, run it after applying a hotfix.
5. EnableServices.ps1 — PowerShell script to be run by the XenDesktop administrator on an XenDesktop Controller.

To upgrade manually

1. Start Studio. The wizard detects what components need to be upgraded and displays the Mandatory upgrade page.
2. Select Manually upgrade this site.
3. The wizard checks for License Server compatibility. Make sure your License Server and license files are compatible with your version of XenDesktop 7.x. Select the confirmation check box and click Continue.
4. The wizard prompts you to backup the Database. When you have done so, select the confirmation check box and click Continue.
   - The wizard generates the manual upgrade scripts that you must run and displays them in a window
   - The Mandatory Upgrade page changes to display a checklist of the manual upgrade steps
5. Make sure you have completed the checklist tasks and click Finish upgrade and return to Common Tasks.

Upgrade core components

The installer automatically upgrades core components such as Delivery Controllers, Studio, and Director if they were previously installed.

1. Log on to the server using a local administrator account and run the installer by inserting the media or mounting the ISO drive for this release, and double-clicking AutoSelect.
2. On the Welcome page, click Start. The wizard detects what components need to be upgraded and displays the Upgrade options page, activating the components you can upgrade.
3. Click Delivery Controller.
4. Accept the license agreement.
5. Review the upgrade steps, click I’m ready to continue and click Next.
6. On the Core Components page review the components available for upgrade.
7. On the Firewall page review the default ports and configure firewall rules.
8. On the Upgrade page review the prerequisites to be installed and the components to be upgraded and then click Upgrade.
9. On the Finish Upgrade page one of the following messages appears upon completion:
   - Success — Upgrade successful appears when the upgrade completes without errors.
   - Failed — The Upgrade failed appears with a list of failed components. Click Why did this fail to review what you must do to fix the problem. Other components that installed successfully are retained; you do not need to reinstall them.
10. Click Finish to complete the upgrade.
11. After completing the XenDesktop upgrade, upgrade machine catalogs as described in Upgrade a machine catalog and
Delivery Groups as described in Upgrade a Delivery Group.

Upgrade a Site with components deployed on the same machine

The following figure shows the high-level processes involved when upgrading a Site in which all components are deployed on the same machine.

Perform the upgrade procedures in the following order:

1. Make sure that your Subscription Advantage date for licenses is no earlier than 2013.0522.
2. Back up the Controller databases as described in How to backup and Restore your XenDesktop Database.
3. Optionally back up templates and upgrade Hypervisor.
4. Upgrade PVS servers and agents as described in Upgrade Provisioning Services.
5. Upgrade the core components as described in Upgrade core components.
6. Upgrade VDAs as described in Upgrade the Virtual Delivery Agent.
7. Upgrade the database using Studio as described in Upgrade Databases using Studio.
8. Upgrade the remaining Delivery Controllers.
9. After completing the XenDesktop upgrade, upgrade machine catalogs as described in Upgrade a machine catalog and Delivery Groups as described in Upgrade a Delivery Group.

Post upgrade processes
After the upgrade and data migration is complete, you can run environment and configuration tests to make sure that the Site is in functional order.

To test the upgraded site, select Test Site in the Site Configuration pane of the Studio Common Tasks page.
License

Mar 28, 2015

For the complete documentation for the Citrix license server, see Licensing section of Citrix eDocs.

For basic license server information to help you get started with XenDesktop 7, refer to the following topics:

- System Requirements
- Install
- Manage Licensing

What's New

**Citrix Simple License Service** - Enables allocation and installation of license files on a license server using a web page interface. Connect to the Simple License Service locally with the Start menu shortcut or remotely with the Simple License Service URL.

**Citrix Web Services for Licensing** - Replaces the Citrix Licensing Configuration Service. The enhanced Citrix Web Services for Licensing provides similar functionality for newer product consoles, as well as additional functionality.

**IPv6 and IPv4** - This release supports pure IPv4, pure IPv6, and dual-stack deployments that use overlapping IPv4 and IPv6 networks.

Fixed issues

- This release of the Simple License Service displays in English only. [#0310912]

Known issues

This section contains:

- Installation issues
- Other known issues
- Simple License Service
- License files
- Event log viewer messages
- Product-specific issues

Installation issues

- If Citrix Licensing is installed in a clustered environment and Windows Firewall is enabled (the default configuration for Windows 2008), connections can fail. Connecting remotely to the console or checking out licenses works until failover occurs in the cluster. Exception rules for CITRIX.exe and lmadmin.exe are created during installation of Licensing and Simple License Service, but do not work after a cluster failover. To work around this issue, create exceptions for Licensing components on the Exceptions tab of the Windows Firewall panel. Create an exception for each of the following ports: Console Web Server port (default port is 8082); License Server Manager port (default port is 27000); Simple License Service port (default port is 8083); and Vendor Daemon port (default port is 7279). This issue occurs with Windows Server 2008 (32-bit and 64-bit), Windows Server 2008 R2, and Windows Server 2012. For more information, see http://support.microsoft.com/kb/2568645. [#232365]
- At the end of installation, the "License Server Configuration" tool is presented. If you choose to cancel on this page, the
license server does not start. You must reopen the License Server Configuration tool and finish the settings before the license server can start. You can open the tool from C:\Program Files\Citrix\Licensing\LS\resource\LSPostConfigTool.exe. If the License Server Configuration tool fails for any reason, uninstall and reinstall the license server.

- During installation, localized characters in the installation path can cause the installation to fail. Accept the default installation path or enter only ASCII alphabetic letter characters for the installation directory. [#229456]
- When configuring the product-side setting for the license server name, do not use localhost. Though you can use the host name, IP address, or FQDN instead, Citrix recommends you use the FQDN. [#165986]
- When you have the Citrix License Server 11.11.1 installed and then install XenDesktop 5.6, the 30-day free trial license is the only license available to you. Workaround: Accept the trial license and complete the installation. Use Desktop Studio to change the product edition and license model settings after the installation. [#0388512]
- The user list for the License Administration Console and the Citrix Simple License Service web page does not support non-ASCII characters in user/group names. Due to this limitation, on a Russian operating system, the BUILTIN Administrators group is not added to the user list, as it is created with non-ASCII characters. This applies to both fresh installs and upgrades. Any users belonging to the BUILTIN Administrators group in an earlier release of XenDesktop and the Simple License Service will not have access to the License Administration Console or the Simple License Service after an upgrade.

As a workaround, add ASCII-character versions of Russian users/groups names post installation using the License Administration Console interface. Alternatively, install the license server on one of the other supported operating systems. [#0395305]

Other known issues

- If you include a backslash in a locally managed user name (for example, test\), you cannot delete the user. [#0270349]
- Changing the licensing port after licenses are installed might cause the No such product or vendor exists: CITRIX message to appear on the Dashboard instead of the installed licenses. [#0269423]
- When a locally managed user having an administrator role does not exist, a domain administrator can only add domain users or groups as administrators to the license server. To edit and delete domain administrators or groups on the User Configuration page in the License Administration Console, a locally managed user having the Administrator role must exist. [#0263016 and #0269719]
- In the License Administration Console, localized characters in user names and passwords can display with unexpected results. To avoid this issue, use US-ASCII alphabetic letter characters for user names and passwords in the License Administration Console. [#0272738, #0273089, #0156833, #0156839, #0156969, #223870, #242767]
- When you try to start the License Administration Console or the Simple License Service, a blank page might display if the Internet Explorer Enhanced Security Configuration is enabled and the License Administration Console or the Simple License Service is not in the Trusted Sites. Workaround: Disable Internet Explorer Enhanced Security Configuration. [#382429]
- Older Desktop Studio releases (prior to XenDesktop 7 Studio) using this version of the license server do not display license usage information or manage the license server using the Licensing Node. This version of the license server is fully compatible with XenDesktop and can serve licenses for any Citrix product deployment. If you continue to use an older Desktop Studio, use the License Administration Console that ships with the license server to display license usage, manage license server users, and upload licenses. If you are using a version of XenDesktop prior to the XenDesktop 7 and you upgrade from earlier versions of Citrix License Server for Windows to version 11.11.1, you might see the following warning:

Warning: Installing this update removes the Licensing Configuration Service. As a result, all versions prior to this release of Studio provide only limited licensing information.
Simple License Service

- When you start the Simple License Service from the Start menu shortcut, the Chrome browser might become unresponsive. Workaround: Type the Simple License Service URL directly into the Chrome browser. [#0314254]
- If the License Administration Console is disabled in your XenDesktop installation, only administrators can use the Simple License Service. To allow users access to the Simple License Service, enable the License Administration Console with Desktop Studio or the Set-LicLACEnabledState PowerShell cmdlet, and then configure the users. [#0311191]
- If you configure the Simple License Service with a port that Firefox or Chrome blocks, you must add an exception to the browser settings. Note that if you start the Simple License Service from the Start menu shortcut and you have the exception set, Chrome ignores it. [#0313924]
- The firewall rule is not removed if you uninstall the Simple License Service after uninstalling the License Server 11.6.1. Workaround: Uninstall the Simple License Service before uninstalling the License Server 11.6.1. [#0305195]
- When doing a silent installation of the Simple License Service on a 32-bit or 64-bit Windows 2003 or Windows 2008 server with a nonadministrator account, the installation fails and an event log error might not be produced. [#0305202]

License files

- Certain license types are not covered by Subscription Advantage and therefore appear in alerts in the Dashboard of the License Administration Console indicating that the Subscription Advantage date is expired. You can ignore such alerts for any license not covered by Subscription Advantage. This includes Evaluation licenses, Not for Resale licenses, Early Release licenses and Technology Preview licenses. These types of licenses do not need Subscription Advantage and your Citrix products do not stop working when the Subscription Advantage date is expired. You can verify the status of any license for which you receive an alert by clicking the license on the Dashboard. The license information expands to show the license type (such as Technology Preview), the license expiration date, and the Subscription Advantage date. [#231847]
- License Files with multiple lines referencing HOSTNAME= are ignored by the license server. Licenses in these files cannot be checked out. This issue is caused when you download licenses associated to different license server hostnames into the same license file. Any of the following error messages might appear in the Event Log Viewer:
  - Event ID: 724 Description: [2196] Invalid license key.
  - Event ID: 764 (2196) Wrong hostid on SERVER line in license file.
In addition, the License Administration Console displays the following error message: "...Error List Returned...Unknown Host."

To resolve this issue, download again separate license files for each Citrix product tied to different license server names. See article CTX110909 in the Citrix Knowledge Center for additional information.

- After you replace evaluation license files on the license server with new license files, the Citrix product might continue to display the following license expiration message when users log on: "Warning: The following Citrix Product is using an Evaluation license. This license will expire in..."

To resolve this issue:

1. Remove the old evaluation license files from the license server. (See — Delete license files in the Licensing section of Citrix eDocs.)
2. At the license server, restart the Citrix Licensing service.
3. At the Citrix product server, point to a fictional license server and then point the product back to the actual license server. (See your product’s documentation for information about changing these settings.)
4. If the problem persists, restart the product server.
Event log viewer messages

- The following message appears in the Event Log Viewer: (1224) Client/server comm version mismatch. The client and server programs are running potentially incompatible versions of FLEXnet comm software. This is an informational message indicating that a Citrix product licensing client is communicating with a different version of the Citrix License Server. Ignore this message.

- After upgrading the license server, messages such as the following example might appear in the Event Log Viewer (product may vary from this example): A provider, MgmtEventProv, has been registered in the Windows Management Instrumentation namespace Root\Citrix\Management to use the LocalSystem account. This account is privileged and the provider may cause a security violation if it does not correctly impersonate user requests. Ignore the message. The upgrade is successful. [#207927, #183919]

Product-specific issues

Access Essentials / XenApp Fundamentals

- After installing Licensing 11.10 and starting the Quick Start tool, a .NET Framework error message indicates that “The service object for the service “CitrixLicensing” could not be acquired. Probably not installed or there is a permission problem.” This issue occurs in Access Essentials 2.0 or XenApp Fundamentals 3.0. To resolve the issue, uninstall Licensing 11.6.1 Build 10007 and install Licensing 11.6.1 Build 9020, available from the License Server Downloads page. [#232048]

- Before installing the license server version 11.10 on systems running Access Essentials 2.0, you must install Hotfixes AEE200W2K3001 and AEE200W2K3002. Failure to do so might render Access Essentials inoperable.

Citrix Branch Repeater with Windows Server

- If you install the same license twice, the second time may trigger a change in the Send Bandwidth Limit, which will be set to the maximum licensed speed. To solve this issue, do not install the same license twice. If you must do so, verify that the Send Bandwidth Limit is set correctly. [#53894]

Citrix MetaFrame Access Suite 3.0

- Upgrading licensing from Citrix MetaFrame Access Suite 3.0 to a newer version of licensing. During the upgrade process of the license server, the licensing components are detected as installed but there is a prompt to remove them. You cannot upgrade licensing in a Citrix product that has licensing 1.0.0 installed as it is incompatible with the licensing that is used in Citrix products today. Uninstall the licensing components and reinstall the latest version. [#217704]

Citrix XenApp Management Pack

- Details about the Citrix License Server are always blank on the Monitoring tab of the System Center Operations Manager for Citrix Managed Servers. There is no workaround for this issue. [#192159]
System requirements for Citrix Licensing

Mar 12, 2015
Requirements for Licensing for Windows

Citrix Licensing is compatible with the same hardware required to support the compatible operating systems. No additional hardware is required.

Note: The license server does not support multi-homing (two network cards plugged into distinct networks). The License Administration Console manages the license server on the computer on which it is installed. It cannot manage remote license servers. The Simple License Service can install licenses only on the license server where it is installed.

<table>
<thead>
<tr>
<th>Operating Systems</th>
<th>You can install the license server on servers running the following Microsoft operating systems. Citrix recommends that you install the latest Microsoft Service Pack and updates.</th>
</tr>
</thead>
</table>
|                   | ● Windows Server 2008 Family  
                   | ● Windows Server 2008 R2 Family  
                   | ● Windows Server 2012 Family  
                   | ● Windows Server 2012 R2 Family  
                   | ● Windows 7, 32-bit and 64-bit editions  
                   | ● Windows 8, 32-bit and 64-bit editions  
                   | ● Windows 8.1, 32-bit and 64-bit editions  |

| Disk Space Requirements | ● 50 MB for the licensing components  
|                        | ● 2 GB for User/Device licensing |

<table>
<thead>
<tr>
<th>Microsoft .NET Framework Requirements</th>
<th>Microsoft .NET Framework 3.5 is required.</th>
</tr>
</thead>
</table>

| Browsers | ● Internet Explorer Version 8 and 9 in compatibility mode  
|          | ● Mozilla Firefox Version 14.0 and 15.0  
|          | ● Chrome Version 14.0 and 15.0  
|          | ● Safari Version 5.1 |
Manage licensing

Dec 03, 2013

You can use Studio to manage and track licensing, provided the license server is in the same domain as Studio, or in a trusted domain. For information about other licensing tasks, see Licensing in Citrix eDocs. You must be a full license administrator to carry out the tasks described below, except for viewing license information. To view license information in Studio, a XenDesktop administrator must have at minimum the Read Licensing Delegated Administrator permission. These built-in roles have that permission:

- Full Administrator
- Read-Only Administrator

To view license information

1. Select the Configuration node in the Studio left pane.
2. Select Licensing. A summary of license usage and settings for the site is displayed with a list of all the XenDesktop licenses currently installed on the specified license server.

To download a license from Citrix

1. In the Studio left pane, select the Configuration node, and then Licensing.
2. In the Actions pane, select Allocate licenses.
3. In the text box, type the License Access Code, which is supplied in an email from Citrix.
4. Select a product from the displayed list and click Allocate licenses. All the licenses available for that product are allocated and downloaded. Note that once you allocate and download all the licenses for a specific License Access Code, you cannot use that License Access Code again. If you must perform additional transactions with that code, log on to My Account.

To add licenses from a network

Use this procedure to add licenses that are stored on your local computer or somewhere on the network.

1. In the Studio left pane, select the Configuration node, and then Licensing.
2. In the Actions pane, select Add licenses.
3. Browse to a license file and add it to the license server.

To change the license server

1. In the Studio left pane, select the Configuration node, and then Licensing.
2. In the Actions pane, select Change license server.
3. Type the address of the license server to use, then click OK. You must specify the address as name:port, where name can be a DNS, NetBIOS, or IP address. If you do not specify a port number, the default port (27000) is assumed.

To select the type of license to use

When configuring the site, after you specify the license server you are prompted to select the type of license to use. If there are no licenses on the server, the option to use the product for a 30-day trial period without a license is automatically selected.

If there are licenses on the server, their details are displayed and you can select one of them. Alternatively, you can add a
license file to the server and then select that one.

To change the product edition and licensing model

1. In the Studio left pane, select the Configuration node, and then Licensing.
2. In the Actions pane, select Edit product edition.

   Important: You cannot use Studio to configure your site to use your XenApp Platinum licenses. If you select XenDesktop Apps Edition and Concurrent Licensing in the Studio Edit product edition UI, the site is configured to consume XenApp Enterprise licenses. Workaround: Use the PowerShell Set-ConfigSite cmdlet:
   Set-ConfigSite -ProductCode MPS -ProductEdition PLT -LicensingModel Concurrent

3. Update the appropriate options and click OK.

To access the License Administration Console

1. In the Studio left pane, select the Configuration node, and then Licensing.
2. In the middle pane, choose the License Administration Console.

   Either the console appears immediately, or, if the dashboard has been configured to be password-protected, you are prompted for your License Administration Console credentials.

   For details about how to use the console, see The License Administration Console in Citrix eDocs.

To add a licensing administrator

XenDesktop administrators who are also licensing administrators have access to the Administrators tab in the Licensing node.

1. In the Studio left pane, select the Configuration node, and then Licensing.
2. In the middle pane, choose the Administrators tab.
3. In the Actions pane, select Add licensing administrator.
4. Browse to the user you want to add as an administrator and choose the permissions.

To edit or delete a licensing administrator

XenDesktop administrators who are also licensing administrators have access to the Administrators tab in the Licensing node.

1. In the Studio left pane, select the Configuration node, and then Licensing.
2. In the middle pane, choose the Administrators tab and select the administrator who you want to delete or edit. When you select an administrator, the options to Edit licensing administrator (to change the administrator permissions for that administrator) and Delete licensing administrator appear in the Actions pane.

To add a licensing administrator group

Adding an Active Directory Group gives licensing administrator permissions to the users within that group. XenDesktop administrators who are also licensing administrators have access to the Administrators tab in the Licensing node.

1. In the Studio left pane, select the Configuration node, and then Licensing.
2. In the middle pane, choose the Administrators tab.
3. In the Actions pane, select Add licensing administrator group.
4. Browse to the group you want to act as licensing administrators and choose the permissions.
To edit or delete a licensing administrator group

XenDesktop administrators who are also licensing administrators have access to the Administrators tab in the Licensing node.

1. In the Studio left pane, select the Configuration node, and then Licensing.
2. In the middle pane, choose the Administrators tab and select the administrator group you want to delete or edit. When you select a licensing administrator group, the options to Edit licensing administrator group (to change the administrator permissions for that group) and Delete licensing administrator group appear in the Actions pane.
Before you upgrade

Before you upgrade a XenDesktop 5 Site, familiarize yourself with the new concepts and components that require upgrading and their versions, and the sequence for a successful upgrade.

New concepts and features

Compared to previous versions, XenDesktop 7.x has many new features and introduces new concepts.

- For new concepts, terminology, and system requirements, see About XenDesktop and Information for administrators of previous versions of XenDesktop.
- For XenApp administrator information, see Important information for XenApp administrators.

Manage multiple versions of XenApp and XenDesktop

There is no XenApp to XenDesktop upgrade.

The Studio management and Director can monitor and manage only XenDesktop 7.x and XenApp 7.5 sites. The monitoring and management tools do not support past versions of XenDesktop or XenApp.

Citrix recommends that if you continue running deployments of past versions of XenApp or XenDesktop, you run them in parallel with the XenDesktop 7.x site and continue running the management consoles with each release for that site.

For example, in a mixed environment, to continue using Desktop Director 2.1 to monitor XenApp 6.5, ensure that Desktop Director 2.1 is installed on a separate server from Director 7.

Sites with Controllers at version 5.x and Virtual Delivery Agents (VDAs) at version 7.x should remain in that state only temporarily. Ideally, you should complete the upgrade of all components as soon as possible.

Citrix recommends that you use StoreFront to aggregate applications and desktops from the different versions of XenApp and XenDesktop. For details, see the StoreFront documentation.

Upgrade XenDesktop 5 components

These components require upgrading:

- License server and licenses
  Note: Following an upgrade to XenApp Platinum Edition to enable platinum features such as Provisioning Services (PVS), XenDesktop may not be able to use the licenses as expected. This is due to an issue where Studio fails to discover XenApp Platinum licenses. To use XenApp Platinum licenses, use the PowerShell SDK to change the licensing setting as described in Use XenApp Platinum licenses with XenDesktop 7.
- Virtual Desktop Agents for Desktop OS Machines (Windows desktop) that upgrade to XenDesktop 7.x VDAs
- Delivery Controllers and other infrastructure components, such as Director
- Existing database

Important: Ensure that you back up your database as described in How to backup and Restore your XenDesktop Database before performing any upgrade procedures.
VDAs

For VDA upgrade recommendations, see XenDesktop 5 upgrade components.

VDAs on Windows XP or Windows Vista

Some XenDesktop 7.x features are not supported on Windows XP or Windows Vista. If the installer detects a VDA running on a Windows XP or Windows Vista machine, it launches a different installer that installs the latest VDA version that is supported on Windows XP or Windows Vista (Version 5.6 FP1 with certain hotfixes). Although these machines and their Machine Catalogs and Delivery Groups cannot use all XenDesktop 7.x features, they can run in the XenDesktop 7.x Site.

VDAs on Windows XP or Windows Vista do not support the following 7.x features:

- Configuration of App-V applications from Studio.
- Configuration of Receiver Storefront addresses from Studio.
- Automatic support for Microsoft Windows KMS licensing when using Machine Creation Services (MCS). KMS licensing is supported using the procedure documented in http://support.citrix.com/article/CTX128580.
- Full information in Director:
  - Logon times and logon end events impacting the logon duration times in Dashboard, Trends, and User Detail views.
  - Logon duration breakdown details for HDX connection time, authentication time, profile load duration, GPO load duration, logon script duration, and interactive session establishment duration.
  - Some categories of Machine and Connection failures states.
  - Activity Manager in Help Desk and User Details views.

Controllers

You can upgrade the following Controller versions:

- 5.0
- 5.0 Service Pack 1
- 5.5
- 5.6
- 5.6 Feature Pack 1

Director

You can upgrade the following Director versions:

- 1.0
- 1.1
- 2.0
- 2.1

Database

After manually backing up your Site database as described in How to backup and Restore your XenDesktop Database, you upgrade the Database from an upgraded Delivery Controller. This process updates the schema and migrates data. Studio also performs additional data migration steps for the services.

Other components

The installer also upgrades the following components:
Personal vDisk (PVD)

Receiver

- If Receiver for Windows (Receiver.exe) resides on a machine, it is upgraded to Receiver for Windows 4.0
- If Receiver for Windows Enterprise (CitrixReceiverEnterprise.exe) resides on a machine, it is upgraded to Receiver for Windows Enterprise 3.4

Components that require separate upgrade

You need to upgrade the following components outside of the in-place upgrade process:

- Provisioning Services (PVS)
  - Upgrade the Provisioning Services server using the Provisioning Services server rolling upgrade
  - Upgrade the Provisioning Services client using Provisioning Services vDisk versioning

  Important: If you intend to run XenDesktop 7.x and XenDesktop 5.6 sites simultaneously, do not upgrade to Provisioning Services 7.

- Microsoft System Center Virtual Machine Manager — XenDesktop 7.x supports SCVMM 2012 and SCVMM 2012 SP1, while XenDesktop 5.x supports SCVMM 2008 R2 SP1. Upgrade in the following sequence so that XenDesktop can continue to operate without any downtime.
  - All XenDesktop Controllers to XenDesktop 5.6 FP1
  - SCVMM server to SCVMM 2012
  - XenDesktop to 7
  - Upgrade SCVMM server from 2012 to 2012 SP1 (optional)

- For external Web Interface servers, configure StoreFront to provide the desktops formerly provided through Web Interface. See the StoreFront documentation. After upgrading to XenDesktop 7.x, add XenDesktop to your StoreFront deployment.

After upgrading all components, you can optionally use Studio to upgrade Machine Catalogs and Delivery Groups.
XenDesktop 5 upgrade components

Aug 14, 2014
Upgrade components

When you run the installer (AutoSelect), the wizard checks whether certain Site components (such as Delivery Controllers and VDAs) need to be upgraded. If you choose not to upgrade some components during this process, when you run Studio notifies you which components need to be upgraded. You cannot proceed to manage your Site until you upgrade these components.

Depending on your Site, the procedures that you perform and the order in which you perform these procedures may vary.

Important: Back up your databases as described in How to Backup and Restore your XenDesktop Database before performing any upgrade procedures.

Upgrade the License Server

Before upgrading the License Server, make sure the Subscription Advantage date for license files that is compatible with a supported edition of the product.

1. Log on to the server using a local administrator account and run the installer by inserting the media or mounting the ISO drive for this release, and double-clicking AutoSelect.
2. On the Welcome page, click Start. The wizard detects what components need to be upgraded and displays the Upgrade options page, activating the components you can upgrade.
3. Accept the license agreement.
4. If the wizard detects an incompatible license server or license files, you are prompted to upgrade that license component. Upgrade the component and then run the installer again, as described in Upgrade core components.

Upgrade the Virtual Delivery Agent

You can only upgrade Desktop OS (Windows Desktop) Virtual Delivery Agents (VDAs) when upgrading from XenDesktop 5.x to XenDesktop 7.x.

For Remote PC Access deployments, Citrix recommends that you use the command line interface to upgrade the VDA in the PC.

Important: To upgrade a VDA 5.x version to the new version, Citrix recommends installing the new VDA on a clean machine image (this could also be an earlier golden image that does not contain a VDA). If that is not feasible, manually uninstall the 5.x VDA using the Windows feature for removing or changing programs, and then install the new VDA. To install the new VDA, follow the guidance in the installation documentation.

Upgrade core components

The installer automatically upgrades core components such as Delivery Controllers, Studio, and Director if they were previously installed.

1. Log on to the server using a local administrator account and run the installer by inserting the media or mounting the ISO drive for this release, and double-clicking AutoSelect.
2. On the Welcome page, click Start. The wizard detects what components need to be upgraded and displays the Upgrade
options page, activating the components you can upgrade.

3. Click Delivery Controller.
   
   Note: If the program detects the XenDesktop Express edition, you are prompted to obtain and install a license for a supported edition. You cannot continue the upgrade until you obtain and install a license for VDI, Enterprise, or Platinum edition before upgrading.

4. Accept the license agreement.
5. Review the upgrade steps, click I’m ready to continue and click Next.
6. On the Core Components page review the components available for upgrade.
7. On the Firewall page review the default ports and configure firewall rules.
8. On the Upgrade page review the prerequisites to be installed and the components to be upgraded and then click Upgrade.
9. On the Finish Upgrade page one of the following messages appears upon completion:
   
   - Success — Upgrade successful appears when the upgrade completes without errors.
   - Failed — The Upgrade failed appears with a list of failed components. Click Why did this fail to review what you must do to fix the problem. Other components that installed successfully are retained; you do not need to reinstall them.

Select Launch Studio to start Studio when the upgrade completes and click Finish.

Upgrade the Database using Studio

After upgrading the core components, use Studio to upgrade the Database.

Automatically upgrade the Database

1. Start Studio. The wizard detects which components need to be upgraded and displays the Mandatory upgrade page.
2. Select Start the Site upgrade automatically.
3. At the prompt, select I am ready to upgrade.
   
   The wizard displays the upgrade progress. After the upgrade completes, the wizard performs tests. This takes several minutes.

4. In the Site Upgrade Complete window, you can optionally view a data migration report and then click Finish.
5. In the Upgrade successful page, click Finish upgrade and return to the Site overview.

Manually upgrade the Database

To minimize Site downtime during a manual upgrade, it is important that the Studio administrator coordinates closely with the database administrator. This process requires that you run a script that temporarily disables services while the manual upgrade scripts run. Ideally, immediately after these scripts complete, the Studio administrator should enable services and complete the upgrade using Studio.

A manual upgrade requires:

- Compatible License Server and license files
- Database backup
- Running the generated scripts in the following order:

1. DisableServices.ps1 — PowerShell script to be run by the Studio administrator on a Controller
2. UpgradeDatabase.sql — SQL script run by the database administrator using a preferred tool (for example, SQL Server Management Studio)
3. EnableServices.ps1 — PowerShell script to be run by the Studio administrator on a Controller
To upgrade manually

1. Start Studio. The wizard detects what components need to be upgraded and displays the Mandatory upgrade page.
2. Select Manually upgrade this site.
3. The wizard checks for License Server compatibility. Make sure your License Server and license files are compatible with your XenDesktop 7.x version. Select the confirmation check box and click Continue.
4. The wizard prompts you to back up the database. When you have done so, select the confirmation check box and click Continue.
   - The wizard generates the manual upgrade scripts that you must run and displays them in a window
   - The Mandatory Upgrade page changes to display a checklist of the manual upgrade steps
5. Make sure you have completed the checklist tasks and click Finish upgrade and return to Common Tasks.

Upgrade the remaining Delivery Controllers

Upgrade any additional Controllers in your Site.

1. On the Common Tasks page click Upgrade remaining Delivery Controllers. A list of Controllers appears.
2. Perform all the previously described tasks, starting with — To upgrade core components on each Controller.
3. When you have upgraded all Controllers, click I have upgraded remaining Delivery Controllers and click Finish.
4. Close Studio and then reopen Studio to implement the changes.
5. In the Site Configuration section of the Common Tasks page, select Perform registration. Registering the remaining Controllers makes the Controllers and their services available to the Site.
XenDesktop 5 upgrade factors

Jan 09, 2014

Upgrade considerations

- Existing Sites
  - You must use the procedure known as an in-place upgrade. You cannot import or migrate data from a XenDesktop 5 Site to a XenDesktop 7.x Site.
  - Although parallel sites (for example, XenDesktop 5 and XenDesktop 7.1) are allowed, you cannot manage a XenDesktop 5 Site with XenDesktop Studio 7.x. Also, you cannot install XenDesktop Studio 7.x on the same machine as a XenDesktop 5 Studio, unless you intend to upgrade the XenDesktop 5 Site. That is, you cannot run side-by-side Studio installations.
  - Do not upgrade a stand-alone Desktop Studio to Desktop Studio 7.x if you are not ready to use XenDesktop 7.x.
  - You cannot use Studio to configure your site to use your XenApp Platinum licenses. See Manage licensing for information about handling XenApp Platinum licenses with XenDesktop.

- Delivery Controllers — For Sites with one Controller, the Site is inoperable during the upgrade. For Sites with more than one Controller, the Site can continue to operate during the upgrade. The upgrade only causes a brief interruption in establishing new client connections during the final database upgrade steps. There may also be times when the Site has reduced capacity because fewer Controllers are available.

- VDAs — You cannot upgrade Virtual Desktop Agents running on Windows XP or Windows Vista to version 7 Virtual Delivery Agents. You must upgrade these VDAs to the Windows XP or Windows Vista version provided by the installer, or upgrade them to Version 5.6 Feature Pack 1.

- Provisioning Services (PVS)
  - Before upgrading a Site, upgrade any Provisioning Services servers that are installed on management machines in the environment. See Upgrading Provisioning Servers.
  - Provisioning Services 7.x does not support creating new desktops with XenDesktop 5 versions. Therefore, although the existing desktops continue to work, you cannot create new desktops with Provisioning Services until you complete the upgrade of your controllers and database to XenDesktop 7.x. If you intend to upgrade Provisioning Services, keep this limitation in mind when planning your entire site upgrade.
  - If you have XenApp Platinum licenses, you can carry forward Provisioning Services to use with a XenDesktop 7.x App Edition license. To do so, you must configure the correct license settings using the PowerShell SDK cmdlet on the Site's Delivery Controller, as described in Manage licensing.

High level upgrade steps

The following diagrams show the high level steps for XenDesktop 5 upgrade.
Post upgrade processes

After the upgrade and data migration is complete, you can run environment and configuration tests to make sure that the Site is in functional order.

To test the upgraded site, select Test Site in the Site Configuration pane of the Studio Common Tasks page.
Install

Mar 28, 2015

You can install components using a wizard-based graphical interface or a command-line interface, which enables scripted installation. Both methods install most prerequisites automatically.

1. Install the core components: Delivery Controller, Citrix Studio, Citrix Director, Citrix License Server, and Citrix StoreFront.
2. From Studio, create a Site.
3. If you will be using provisioning tools (such as Machine Creation Services or Provisioning Services) to create VMs from a master image, install a VDA for Desktop OS. Otherwise, install a VDA for Desktop OS or a VDA for Server OS directly on each machine.
   For Remote PC Access deployments, Citrix recommends using your existing Electronic Software Distribution (ESD) methods by using the VDA installer's command line interface.
4. After you install components and create a Site, use the guidance in Studio to create machine catalogs and Delivery Groups.

The following tables list tasks to complete and things to consider or be aware of before installing the XenDesktop core components (Delivery Controller, Citrix Studio, Citrix Director, Citrix License Server, StoreFront) and Virtual Delivery Agents (VDAs).

Core component and general installation preparation

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>First:</td>
</tr>
<tr>
<td>● Check the</td>
</tr>
<tr>
<td>— Known issues</td>
</tr>
<tr>
<td>topic for installation issues you might encounter.</td>
</tr>
<tr>
<td>● If you're unfamiliar with XenDesktop components, read the</td>
</tr>
<tr>
<td>— Component overview</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Decide where you will install the XenDesktop components and then prepare the machines and operating systems.</td>
</tr>
<tr>
<td>● Review the</td>
</tr>
<tr>
<td>— System requirements</td>
</tr>
<tr>
<td>topic for supported operating systems and versions for the Controller, Studio, Director, Host, and VDAs. The Citrix StoreFront and the Citrix License Server requirements topics specify their supported platforms.</td>
</tr>
<tr>
<td>You can install the core components on the same server or on different servers. For example, to manage a smaller deployment remotely, you can install Studio on a different machine than the server where you installed the Controller. To accommodate future expansion, consider installing components on separate servers; for example, install the License Server and Director on different servers.</td>
</tr>
<tr>
<td>● Be sure that each operating system has the latest updates.</td>
</tr>
<tr>
<td>● Be sure that all machines have synchronized system clocks. Synchronization is required by the Kerberos infrastructure that secures communication between the machines.</td>
</tr>
<tr>
<td>● XenDesktop components are installed in C:\Program Files\Citrix by default. You can specify a different location during installation, but it must have execute permissions for network service.</td>
</tr>
</tbody>
</table>

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.72
Most component prerequisites are installed automatically; however, check the System requirements topic for exceptions.

Decide where to install the SQL Server software for the Site Configuration Database.
- By default, SQL Server 2012 Express is installed automatically on the server when you install the Controller. Alternatively, you can separately install a supported SQL Server version on that server or on a different server. In such cases, the SQL Server software does not need to be installed before you install the XenDesktop components, but it must be installed before you create the XenDesktop Site.
- Review the database considerations in the Plan topics, and set up any supported redundancy infrastructure.

Important: Windows authentication is required between the Controller and the database.

Decide how you want ports opened.

XenDesktop opens the following ports automatically if the Windows Firewall Service is running, even if the firewall is not enabled. You can disable this default action and open the ports manually if you use a third-party firewall or no firewall, or if you just prefer to do it yourself.
- Controller: TCP 80, 443
- Director: TCP 80, 443
- License Server: TCP 7279, 8082, 8083, 27000
- StoreFront: TCP 80, 443

Tip: For complete port information, see CTX101810.

Configure your Active Directory domain. See the Microsoft documentation for instructions.
- In addition to being a domain user, you must be a local administrator on the machines where you are installing core components.
- Do not attempt to install any components on a domain controller.
- For information about multiple forest support, see Deploy in a multiple forest Active Directory environment.

When you install the License Server, that user account is automatically made a full administrator on the license server.

When you install Director, decide if you want the shadowing feature of Director Server.

Good to know:
- If a component does not install successfully, the process stops with an error message. Components that installed successfully are retained; you do not need to reinstall them.
- Studio starts automatically after it is installed. You can disable this action during installation.
- When you create objects before, during, and after installation, it is best practice to specify unique names for each object (for example networks, groups, catalogs, resources).

VDA installation preparation
## Description

If you will be installing a VDA for Windows Desktop OS, decide if you want to install the HDX 3D Pro version.

The HDX3D Pro feature delivers desktops and applications that perform best with a GPU for hardware acceleration. For more information, see the HDX 3D Pro documentation.

### Decide how you will use the VDA.

XenDesktop assumes that you will use a master image of the installed VDA with Machine Creation Services or Provisioning Services to create other virtual machines. You can override this default if you want to install the VDA on an existing machine.

### Decide if you want to install Citrix Receiver for Windows (CitrixReceiver.exe).

You can disable this default action.

### Decide how you want ports opened.

XenDesktop opens ports automatically if the Windows Firewall Service is running, even if the firewall is not enabled. You can disable this default action and open the ports manually if you use a third-party firewall or no firewall, or if you just prefer to do it yourself.

- **Controller:** TCP 80, 1494, 2598, 8008
  - For communication between user devices and virtual desktops, configure inbound TCP on ports 1494 and 2598 as port exceptions. For security, Citrix recommends that you do not use these registered ports for anything other than the ICA protocol and the Common Gateway Protocol.
  - For communication between Controllers and virtual desktops, configure inbound port 80 as a port exception.
  - **Windows Remote Assistance:** TCP 3389
    - Windows opens this port automatically if the feature is enabled, even if you choose to open the ports manually.
- **Real-Time Audio Transport:** UDP 16500-16509

**Tip:** For complete port information, see [CTX101810](https://docs.citrix.com).

### Decide how the locations of installed Controllers will be specified.

- **Manually,** by entering the Fully Qualified Domain Name (FQDN) of the Controller. Although you can specify a Controller that is not currently in the domain, a VDA can connect only to a Controller in the domain. Also, you can test the connection only for Controllers in the domain.
- **Using Active Directory,** if the Controller is in the domain.
- **Allowing Machine Creation Services to specify the Controller.**
- **Later,** by rerunning the installer, using Citrix policies, setting registry values, or using Active Directory OUs.

Citrix Group Policy settings that specify Controller locations will override settings provided during installation.

After you initially specify the Controller location, you can use the auto-update feature to update VDAs when additional Controllers are installed. For more information about methods for specifying Controllers, see [Manage your Delivery Controller environment](https://docs.citrix.com).

---

https://docs.citrix.com

© 1999-2017 Citrix Systems, Inc. All rights reserved.
### Decide if you want to use the following features:

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Optimize performance:</strong> When this feature is enabled, the optimization tool is used for VDAs running in a VM on a hypervisor. VM optimization includes disabling offline files, disabling background defragmentation, and reducing event log size. For more information, see <a href="https://docs.citrix.com">CTX125874</a>. Do not enable this option if you will be using Remote PC Access. Default = enabled.</td>
</tr>
<tr>
<td><strong>Windows Remote Assistance:</strong> When this feature is enabled, Windows Remote Assistance is used with the user shadowing feature of Director, and Windows automatically opens TCP port 3389 in the firewall, even if you choose to open firewall ports manually. Default = enabled.</td>
</tr>
<tr>
<td><strong>Real-Time Audio Transport for audio:</strong> When this feature is enabled, UDP is used for audio packets, which can improve audio performance. Default = enabled.</td>
</tr>
<tr>
<td><strong>Personal vDisk:</strong> (Available only when installing a VDA for Windows Desktop OS on a VM.) When this feature is enabled, Personal vDisks can be used with a master image. For more information, see — <a href="https://docs.citrix.com">Manage Personal vDisks</a>. Default = disabled.</td>
</tr>
</tbody>
</table>

### Good to know:

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>The installer automatically detects your operating system and allows you to install only the VDA type supported on that system:</strong> VDA for Windows Server OS or VDA for Windows Desktop OS.</td>
</tr>
<tr>
<td><strong>Profile management is installed during VDA installation.</strong></td>
</tr>
<tr>
<td><strong>When you install the VDA, a new local user group called Direct Access Users is automatically created. On a VDA for Windows Desktop OS, this group applies only to RDP connections; on a VDA for Windows Server OS, this group applies to ICA and RDP connections.</strong></td>
</tr>
<tr>
<td><strong>When you install a VDA for Windows Server OS, Remote Desktop Services role services are automatically installed and enabled (if they are not already installed and enabled).</strong></td>
</tr>
<tr>
<td><strong>For Remote PC Access configurations, install the VDA for Windows Desktop OS on each physical office PC that users will access remotely.</strong></td>
</tr>
<tr>
<td><strong>Do not install a VDA for Windows Server OS on a server that has XenApp 6.5 or earlier installed.</strong></td>
</tr>
</tbody>
</table>
Install using the graphical interface

Mar 28, 2015

Note: Before beginning any installation, review and complete the tasks in Prepare to install.
Launch the installer graphical interface:
1. Download the product package and unzip it. Optionally, burn a DVD of the ISO file.
2. Log on to the server where you are installing the components, using a local administrator account.
3. Insert the DVD in the drive or mount the ISO file. If the installer does not launch automatically, double-click the AutoSelect application or the mounted drive.
4. Select the component you want to install:
   - If you're just getting started, click Delivery Controller. From there, you can install the Delivery Controller and optionally, Studio, Director, License Server, and StoreFront on the same server.
   - If you've already installed some components and want to extend your deployment, click the component you want to install from the right column. This column offers core components and the Universal Print Server, which you can install on your print server.
   - To install a Virtual Delivery Agent (VDA), click the available VDA entry - the installer knows which one is right for the operating system where you're running the installer.

Later, if you want to customize a VDA that you've already installed:
1. From the Windows feature for removing or changing programs, select Citrix Virtual Delivery Agent <version-number>, then right-click and select Change.
2. Select Customize Virtual Delivery Agent Settings. When the installer launches, you can change the Controller addresses, TCP/IP port to register with the Controller (default = 80), or whether to automatically open Windows Firewall port exceptions.
Install using the command line

Feb 03, 2014

Use the command line interface to:

- Install one or more core components: Delivery Controller, Citrix Studio, Citrix Director, License Server, and StoreFront.
- Install a Virtual Delivery Agent (VDA) on a master image or on a virtual or physical machine.

You can also customize scripts provided on the media, then use them to install and remove VDAs in Active Directory; see Install or remove Virtual Delivery Agents using scripts in Active Directory.

- Customize a previously-installed VDA.
- Install a Universal Print Server, which provisions network session printers. (The Controller already has the Universal Print Server functionality; you need only install the Universal Print Server on the print servers in your environment.)

You can also remove components from this version that you previously installed, using the /remove or /removeall options. For more information, see Remove components.

To see command execution progress and return values, you must be the original administrator or use 'Run as administrator.' For more information, see Microsoft command documentation.

Important: Before beginning an installation, read and complete the tasks in Prepare to install.

From the \x64\XenDesktop Setup directory on the media, run the XenDesktopServerSetup.exe command. The following table describes command options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/help or /h</td>
<td>Displays command help.</td>
</tr>
<tr>
<td>/quiet</td>
<td>No user interface appears during the installation. The only evidence of the installation process is in Windows Task Manager. If this option is omitted, the graphical interface launches.</td>
</tr>
<tr>
<td>/logpath path</td>
<td>Log file location. The specified folder must already exist; the installer does not create it. Default = &quot;%TEMP%\Citrix\XenDesktop Installer&quot;</td>
</tr>
<tr>
<td>/noreboot</td>
<td>Prevents a restart after installation. (For most core components, a restart is not enabled by default.)</td>
</tr>
<tr>
<td>/remove</td>
<td>Removes the core components specified with the /components option. For more information about removing components, see Remove components.</td>
</tr>
<tr>
<td>/removeall</td>
<td>Removes all installed core components. For more information about removing components, see Remove components.</td>
</tr>
</tbody>
</table>
Opens all ports in the Windows firewall needed by components being installed, if the Windows Firewall Service is running, even if the firewall is not enabled. If you are using a third-party firewall or no firewall, you must manually open the ports.

/(required) Comma-separated list of components to install or remove. Valid values are:
- CONTROLLER - Controller
- DESKTOPSTUDIO - Studio
- DESKTOPDIRECTOR - Director
- LICENSESERVER - Citrix Licensing
- STOREFRONT - StoreFront

If this option is omitted, all components are installed (or removed, if the /remove option is also specified).

Existing empty directory where components will be installed. Default = C:\Program Files\Citrix.

Directory that holds temporary files during installation. Default = C:\Windows\Temp.

Prevents installation of Microsoft SQL Server Express on the server where you are installing the Controller. If this option is omitted, SQL Server Express will be installed.

(Valid only when installing Director) Enables the Windows Remote Assistance feature. Windows Remote Assistance is always installed when you install Director; you enable or disable the feature by including or omitting this option in the command line.

For example, the following command installs a Controller, Studio, Citrix Licensing, and SQL Server Express on the server. Ports in the Windows firewall that are required for Controller and License Server communications will be opened automatically.

\x64\XenDesktop Setup\XenDesktopServerSetup.exe /components controller,desktopstudio,licenseserver /configure_firewall

Note: For information about installing an earlier Virtual Desktop Agent version on Windows XP or Windows Vista systems, see Install an earlier Virtual Desktop Agent on Windows XP or Windows Vista.

When installing a VDA for use with Remote PC Access, specify only options that are valid on physical machines (not VMs or master images) and for VDAs for Windows Desktop OS.

From the \x64\XenDesktop Setup directory on the product media, run the XenDesktopVdaSetup.exe command. The following table describes command options. Unless otherwise noted, options apply to physical and virtual machines, and to VDAs for Windows Desktop OS and VDAs for Windows Server OS.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/h or /help</td>
<td>Displays command help.</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>/logpath path</td>
<td>Log file location. The specified folder must already exist; the installer does not create it. Default = &quot;%TEMP%\Citrix\XenDesktop Installer&quot;</td>
</tr>
<tr>
<td>/noreboot</td>
<td>Prevents a restart after installation. The VDA will not be fully available for use until after a restart.</td>
</tr>
<tr>
<td>/remove</td>
<td>Removes the components specified with the /components option.</td>
</tr>
<tr>
<td>/removeall</td>
<td>Removes all installed VDA components.</td>
</tr>
<tr>
<td>/reconfig</td>
<td>Customizes previously-configured VDA settings when used with the /portnumber, /controllers, or /enable_hdx_ports options. If you specify this option without also specifying the /quiet option, the graphical interface for customizing the VDA launches.</td>
</tr>
<tr>
<td>/portnumber port</td>
<td>(Valid only if the /reconfig option is specified.) Port number to enable for communications between the VDA and the Controller. The previously-configured port is disabled, unless it is port 80.</td>
</tr>
</tbody>
</table>
| /components component[,component] | Comma-separated list of components to install or remove. Valid values are:  
  - VDA - installs the VDA  
  - PLUGINS - installs the Citrix Receiver for Windows (CitrixReceiver.exe)  
If this option is omitted, all components are installed. |
<p>| /installdir directory | Existing empty directory where components will be installed. Default = c:\Program Files\Citrix. |
| /tempdir directory | Directory to hold temporary files during installation. (This option is not available in the graphical interface.) Default = c:\Windows\Temp. |
| /site_guid guid | Globally Unique Identifier of the site Active Directory Organizational Unit (OU). This associates a virtual desktop with a Site when you are using Active Directory for discovery (auto-update is the recommended and default discovery method). The site GUID is a site property displayed in Studio. Do not specify both the /site_guid and /controllers options. |
| /controllers &quot;controller&quot; | Space-separated Fully Qualified Domain Names (FQDNs) of Controllers with which the... |</p>
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[controller][...]</td>
<td>VDA can communicate, enclosed in quotation marks. Do not specify both the /site_guid and /controllers options.</td>
</tr>
<tr>
<td>/xa_server_location url</td>
<td>URL of the server for Windows server applications.</td>
</tr>
<tr>
<td>/enable_remote_assistance</td>
<td>Enables Windows Remote Assistance for use with Director. If you specify this option, Windows opens TCP port 3389 in the firewall, even if you omit the /enable_hdx_ports option.</td>
</tr>
<tr>
<td>/enable_hdx_ports</td>
<td>Opens ports in the Windows firewall required by the Controller and features you specified (Windows Remote Assistance, real-time transport, and optimize), if the Windows Firewall Service is detected, even if the firewall is not enabled. If you are using a different firewall or no firewall, you must configure the firewall manually.</td>
</tr>
<tr>
<td>/optimize</td>
<td>Enables optimization for VDAs running in a VM on a hypervisor. VM optimization includes disabling offline files, disabling background defragmentation, and reducing event log size. Do not specify this option for Remote PC Access. For more information about the optimization tool, see CTX125874.</td>
</tr>
<tr>
<td>/baseimage</td>
<td>(Valid only when installing a VDA for Windows Desktop OS on a VM.) Enables the use of Personal vDisks with a master image. For more information, see Manage Personal vDisks.</td>
</tr>
<tr>
<td>/enable_hdx_3d_pro</td>
<td>Installs the VDA for HDX 3D Pro. For more information, see the HDX 3D Pro documentation.</td>
</tr>
<tr>
<td>/enable_real_time_transport</td>
<td>Enables or disables use of UDP for audio packets (Real-Time Audio Transport for audio). Enabling this feature can improve audio performance. Recommended for Remote PC Access. (Include the /enable_hdx_ports option if you want the UDP ports opened automatically if the Windows Firewall Service is detected.)</td>
</tr>
<tr>
<td>/masterimage</td>
<td>(Valid only when installing a VDA on a VM.) Sets up the VDA as a master image.</td>
</tr>
<tr>
<td>/isvirtual</td>
<td>(Valid only when installing a VDA on a VM.) Overrides detection by the installer of a physical machine, where BIOS information passed to VMs makes them appear as physical machines.</td>
</tr>
<tr>
<td>/nodesktopexperience</td>
<td>(Valid only when installing a VDA for Windows Server OS.) Prevents enabling of the Enhanced Desktop Experience feature. This feature is also controlled with the Enhanced Desktop Experience Citrix policy setting.</td>
</tr>
</tbody>
</table>
**Option** | **Description**
---|---
/noctrixwddm | (Valid only on Windows 7 machines that do not include a WDDM driver) Disables installation of the Citrix WDDM driver.

/servervdi | Installs a VDA for Windows Desktop OS on a supported Windows Server. Omit this option when installing a VDA for Windows Server OS on a Windows Server. Before using this option, see Server VDI.

/installwithsecurebootenabled | Allows VDA installation when Secure Boot is enabled. If this option is omitted, a warning displays that Secure Boot must be disabled to successfully install a VDA.

The following command installs a VDA for Windows Desktop OS and Citrix Receiver to the default location on a VM. This VDA will be used as a master image. The VDA will register initially with the Controller on the server named 'Contr-Main' in the domain 'mydomain,' and will use Personal vDisks, the optimization feature, and Windows Remote Assistance.

```
x64\XenDesktop Setup\XenDesktopVdaSetup.exe /quiet /components vda,plugins /controllers "Contr-Main.mydomain.local" /enable_hdx_ports /optimize /masterimage /baseimage /enable_remote_assistance
```

The following command installs a VDA for Windows Desktop OS and Citrix Receiver to the default location on an office PC that will be used with Remote PC Access. The machine will not be restarted after the VDA is installed; however, a restart is required before the VDA can be used. The VDA will register initially with the Controller on the server named 'Contr-East' in the domain 'mydomain,' and will use UDP for audio packets. HDX ports will be opened if the Windows Firewall service is detected.

```
x64\XenDesktop Setup\XenDesktopVdaSetup.exe /quiet /components vda,plugins /controllers "Contr-East.mydomain.local" /enable_hdx_ports /enable_real_time_transport /noreboot
```

After you install a VDA, you can customize several settings. From the \x64\XenDesktop Setup directory on the product media, run the XenDesktopVdaSetup.exe command, using one or more of the following options, which are described above.

- /reconfigure - this option is required when customizing a VDA
- /h or /help
- /quiet
- /noreboot
- /controllers
- /portnumber port
- /enable_hdx_ports

Run one of the following commands on each print server:

- On a supported 32-bit operating system: From the \x86\Universal Print Server\ directory on the Citrix installation media, run UpsServer_x86.msi.
- On a supported 64-bit operating system: From the \x64\Universal Print Server\ directory on the Citrix installation media, run UpsServer_x64.msi.
Install VDAs using scripts in Active Directory

Feb 06, 2014

The installation media contains sample scripts that install, upgrade, or remove Virtual Delivery Agents (VDAs) for groups of machines in Active Directory. You can also apply the scripts to individual machines, and use them to maintain master images used by Machine Creation Services and Provisioning Services.

Required access:
- The scripts need Everyone Read access to the network share where the VDA installation command, XenDesktopVdaSetup.exe, is located.
- Logging details are stored on each local machine. If you also want to log results centrally for review and analysis, the scripts need Everyone Read and Write access to the appropriate network share.

To check the results of running a script, examine the central log share. Captured logs include the script log, the installer log, and the MSI installation logs. Each installation or removal attempt is recorded in a time-stamped folder. The folder title indicates if the operation was successful with the prefix PASS or FAIL. You can use standard directory search tools to quickly find a failed installation or removal in the central log share, rather than searching locally on the target machines. For more information, see the Troubleshooting section below.

Important: Before beginning any installation, read and complete the tasks in Prepare to install.

1. Obtain the sample script InstallVDA.bat from \Support\AdDeploy\ on the installation media. Citrix recommends that you make a backup of the original script before customizing it.

2. Edit the script:
   - Specify the version of the VDA to install: SET DESIREDVERSION. For example, version 7 can be specified as 7.0; the full value can be found on the installation media in the ProductVersion.txt file (such as 7.0.0.3018); however, a complete match is not required.
   - Specify the network share location from which the installer will be invoked. Point to the root of the layout (the highest point of the tree): the appropriate version of the installer (32-bit or 64-bit) will be called automatically when the script runs. For example: SET DEPLOYSHARE=\fileserver1\share1.
   - Optionally, specify a network share location for storing centralized logs. For example: SET LOGSHARE=\fileserver1\log1.
   - Specify VDA configuration options as described in Install using the command line. The /quiet and /noreboot options are included by default in the script and are required: SET COMMANDLINEOPTIONS=\quiet /noreboot.

3. Using Group Policy Startup Scripts, assign the script to the OU in Active Directory where your machines are located. This OU should contain only machines on which you want to install the VDA. When the machines in the OU are restarted, the script runs on all of them, installing a VDA on each machine that has a supported operating system.

1. Obtain the sample script UninstallVDA.bat from \Support\AdDeploy\ on the installation media. Citrix recommends that you make a backup of the original script before customizing it.

2. Edit the script:
   - Specify the version of the VDA to remove: SET CHECK_VDA_VERSION. For example, version 7 can be specified as 7.0; the full value can be found on the installation media in the ProductVersion.txt file (such as 7.0.0.3018); however, a complete match is not required.
   - Optionally, specify a network share location for storing centralized logs.
3. Using Group Policy Startup Scripts, assign the script to the OU in Active Directory where your machines are located. This OU should contain only machines from which you want to remove the VDA. When the machines in the OU are restarted, the script runs on all of them, removing a VDA from each machine.

The script generates internal log files that describe script execution progress. The script copies a Kickoff_VDA_Startup_Script log to the central log share within seconds of starting the deployment to the machine, so that you can verify that the overall process is working. If this log is not copied to the central log share as expected, you can troubleshoot further by inspecting the local machine: the script places two debugging log files in the %temp% folder on each machine, for early troubleshooting:

- Kickoff_VDA_Startup_Script_<DateTimeStamp>.log
- VDA_Install_ProcessLog_<DateTimeStamp>.log

Review the content of these logs to ensure that the script is:
- Running as expected.
- Properly detecting the target operating system.
- Correctly configured to point to the ROOT of the DEPLOYSHARE share (contains the file named AutoSelect.exe).
- Capable of authenticating to both the DEPLOYSHARE and LOG shares.
Create a Site

Feb 06, 2014

A Site is the name you give to a product deployment. It comprises the Delivery Controllers and the other core components, VDAs, Host connections (if used), plus the machine catalogs and Delivery Groups you create and manage. A Site does not necessarily correspond to a geographical location, although it can. You create the Site after you install the components and before creating machine catalogs and Delivery Groups.

The following table describes the tasks to complete and things to consider or be aware of before starting the Site creation wizard in Studio.

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decide which type of Site you will create:</td>
</tr>
<tr>
<td>- Application and desktop delivery Site - When you choose to create an application and desktop delivery Site, you can further choose to create a full deployment Site (recommended) or a basic Site. (Basic Sites are usually created by advanced users.)</td>
</tr>
<tr>
<td>- Remote PC Access Site - Allows designated users to remotely access their office PCs through a secure connection.</td>
</tr>
</tbody>
</table>

If you create a full deployment now, you can add a Remote PC Access deployment later. Conversely, if you create a Remote PC Access deployment now, you can add a full deployment later.

Site creation includes creating the Site Configuration database. Make sure the SQL Server software is installed before you create a Site.

To create the database, you must be a local administrator and a domain user. You must also either have SQL Server permissions, or you can generate scripts to give to your database administrator to run.

- Permissions – you need the following permissions when setting up the database; the permissions can be explicitly configured or acquired by Active Directory group membership:

<table>
<thead>
<tr>
<th>Operation</th>
<th>Purpose</th>
<th>Server role</th>
<th>Database role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database creation</td>
<td>Create suitable empty database</td>
<td>dbcreator</td>
<td></td>
</tr>
<tr>
<td>Schema creation</td>
<td>Create all service-specific schemas and add first Controller to Site</td>
<td>securityadmin</td>
<td>db_owner</td>
</tr>
<tr>
<td>Add Controller</td>
<td>Add Controller (other than the first) to the Site</td>
<td>securityadmin</td>
<td>db_owner</td>
</tr>
<tr>
<td>Add Controller (mirror server)</td>
<td>Add Controller login to the database server currently in the mirror role of a mirrored database</td>
<td>securityadmin</td>
<td>db_owner</td>
</tr>
<tr>
<td>Schema update</td>
<td>Apply schema updates or hotfixes</td>
<td>db_creator</td>
<td></td>
</tr>
</tbody>
</table>
While technically more restrictive, in practice, the securityadmin server role should be treated as equivalent to the sysadmin server role.

When using Desktop Studio to perform these operations, the user account must be a member of the sysadmin server role.

If your Studio user credentials do not include these permissions, you are prompted for SQL Server user credentials.

- **Scripts** - If your database server is locked down and you do not have the required SQL Server permissions, a database script can be generated that your database administrator can use to set up the database. A second script is also generated if your database administrator has set up database mirroring. The script generation feature is available in the Site creation wizard – after you request script generation, you give the generated scripts to your database administrator to run on the database server (and the mirrored database, if needed). After the script is executed and the database is successfully created, you can finish creating the Site.

Consider if you will use the 30-day free trial license that allows you to add license files later, or if you will use existing licenses. You can add or download license files from within the Site creation wizard.

Set up your virtualization resource infrastructure, if you are using one.

If you use XenServer:
- You must provide the credentials for a VM Power Admin or higher-level user.
- Citrix recommends using HTTPS to secure communications with XenServer. To use HTTPS, you must replace the default SSL certificate that was installed on XenServer with a certificate from a trusted authority; see [CTX128656](https://docs.citrix.com).
- You can configure high availability if it is enabled on the XenServer.
- Citrix recommends that you select all servers in the pool to allow communication with XenServer if the pool master fails.

If you use VMware, see [VMware and virtual machine management](https://docs.citrix.com).

Decide if you will use Machine Creation Services (MCS) or other tools to create VMs.

Decide if you will use shared or local storage. Shared storage is available through the network. If you use shared storage, you can enable the use of IntelliCache to reduce load on the storage device. For information about IntelliCache, see [Use IntelliCache with XenDesktop](https://docs.citrix.com).

Decide if you will use Personal vDisks and whether they will use shared or local storage. Personal vDisks can use the same or different storage as the VMs. For more information, see [Personal vDisks](https://docs.citrix.com).

Decide if you will use App-V publishing. Configure those resources, if needed. For more information, see [Configure App-V resources](https://docs.citrix.com).

Good to know:
- Creating a Remote PC Access Site creates a machine catalog named Remote PC Access Machines, and a Delivery Group named Remote PC Access Desktops.
When you create a Remote PC Access Site, you must specify users or user groups; there is no default action that automatically adds all users.

- The user who creates a Site becomes a Full Administrator; for more information, see Delegated Administration.
- When an empty database is created, it has default attributes except:
  - The collation sequence is set to Latin1_General_100_CI_AS_KS (where Latin1_General varies, depending on the country, for example Japanese_100_CI_AS_KS). If this collation setting is not specified during database creation, subsequent creation of the service schemas within the database will fail, and an error similar to "<service>: schema requires a case-insensitive database" appears. (When a database is created manually, any collation sequence can be used, provided it is case-sensitive, accent-sensitive, and kanatype-sensitive; the collation sequence name typically ends with _CI_AS_KS.)
  - The recovery mode is set to Simple. For use as a mirrored database, change the recovery mode to Full.
- When you create the Site Configuration Database, it also stores configuration changes recorded by the Configuration Logging Service, plus trend and performance data that is used by the Monitoring Service and displayed by Citrix Director. If you use those features and store more than seven days of data, Citrix recommends that you specify different locations for the Configuration Logging Database and the Monitoring Database (known as the secondary databases) after you create a Site. See Change secondary database locations.

Start Studio, if it is not already open. After you choose to create a Site from the center pane, specify the following:

- The type of Site and the Site name.
- Database information. If you use a database server that is installed on a different server, enter the database server and name:

<table>
<thead>
<tr>
<th>Database type</th>
<th>What to enter</th>
<th>With this database configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standalone or mirror</td>
<td>servername</td>
<td>The default instance is used and SQL Server uses the default port.</td>
</tr>
<tr>
<td></td>
<td>servername\INSTANCENAME</td>
<td>A named instance is used and SQL Server uses the default port.</td>
</tr>
<tr>
<td></td>
<td>servername, port-number</td>
<td>The default instance is used and SQL Server uses a custom port.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(The comma is required.)</td>
</tr>
<tr>
<td>Other</td>
<td>cluster-name</td>
<td>A clustered database.</td>
</tr>
<tr>
<td></td>
<td>availability-group-listener</td>
<td>An Always-On database.</td>
</tr>
</tbody>
</table>

After you click Next and are alerted that the services could not connect to a database, indicate that you want Studio to create it. If you do not have permission to edit the database, use Generate database script. The scripts must be run before you can finish creating the Site.

- License Server address in the form name:[port] where name is a Fully Qualified Domain Name (FQDN), NetBIOS, or IP address; FQDN is the recommended format. If you omit the port number, the default is 27000. You cannot proceed until a successful connection is made to the license server.
- Connection information to your virtualization resource (type, address, and credentials). If you are not using a resource, or if you will use Studio to manage user desktops hosted on dedicated blade PCs, select the Host type None.
If you are using a Host, you also select a network, specify storage information, and indicate if you will use Personal vDisk.

If the specified XenServer Host supports vGPU, you can also select a GPU type and group, or passthrough. The display indicates if the selection has dedicated GPU resources.

- App-V management and App-V publishing server information.
- User and machine accounts information, if you are creating a Remote PC Access Site.
  - User information. Click Add Users. Select users and user groups, and then click Add users.
  - Machine accounts information. Click Add machine accounts. Select machine accounts, and then click Add machine accounts. Click Add OUs. Select the domain and Organizational Units, and indicate if items in subfolders should be included. Click Add OUs.

Configuration and environment tests run automatically when you create a Site. You can view an HTML report of the results. You can also run the tests on demand:
1. From Studio, click the Studio (<Site-name>) entry at the top of the left pane.
2. In the center pane, click Test configuration.
Install a VDA on earlier systems

Apr 20, 2015

The Virtual Delivery Agents (VDAs) with this version number are not supported on Windows XP or Windows Vista systems. Additionally, some of the features in this release are not supported on those operating systems. To use the full functionality in this release, Citrix recommends you replace Windows XP or Windows Vista systems with Windows 7 or Windows 8, then install a Virtual Delivery Agent from this release.

To accommodate cases when you must install an earlier version of the Virtual Desktop Agent to create a Machine Catalog and a Delivery Group to host Windows XP or Windows Vista clients, or to create a Master Image to deliver Windows XP or Windows Vista desktops, the media for this release includes earlier Virtual Desktop Agent software that is supported on those systems (version 5.6 FP1). If the installer detects a Windows XP or Windows Vista system, it launches a different installer that deploys the version 5.6 FP1 Virtual Desktop Agent.

- You cannot install core components (for example, Controller, Studio, Director, StoreFront, Citrix License Server) on a Windows XP or Windows Vista system.
- Remote PC Access is not supported on Windows Vista systems.

Note: When installing the earlier version of a Virtual Desktop Agent on a 32-bit XP machine, either use the installation media locally or copy the media to the local drive. Do not attempt to install the earlier version from a network share or a mapped drive.

1. On the Windows XP or Windows Vista machine, launch the installer for this release and select Virtual Delivery Agent for Windows Desktop OS.
2. When the installer for the Virtual Desktop Agent version 5.6 FP1 launches, select Advanced Install and follow the on-screen instructions. For details about what you specify when installing the earlier version, search for Installing the Virtual Desktop Agent in the archived XenDesktop 5 documentation.
3. After installing the Virtual Desktop Agent, install any additional hotfixes for that release.

From the \x64\XP XenDesktop Setup or \x86\XP XenDesktop Setup directory on the installation media, run the XenDesktopVdaSetup.exe command. For details about what you specify when installing the earlier version, search for XenDesktopVdaSetup.exe in the archived XenDesktop 5 documentation.

After installing the Virtual Desktop Agent, install any additional hotfixes for that release.
Remove components

Sep 06, 2013

To remove XenDesktop components, Citrix recommends using the Windows feature for removing or changing programs. Alternatively, you can remove components using the command line, or a script on the installation media.

When you remove components, prerequisites are not removed, and firewall settings are not changed. When you remove a Controller, the SQL Server software and the databases are not removed.

Before removing a Controller, remove it from the Site; see To remove a Controller. Before removing Studio or Director, Citrix recommends closing them.

If you upgraded a Controller from an earlier XenDesktop deployment that included Web Interface, you must remove the Web Interface component separately; you cannot use the XenDesktop installer to remove Web Interface.

From the Windows feature for removing or changing programs:
- To remove a Controller, Studio, Director, License Server, or StoreFront, select Citrix XenDesktop <version>, then right-click and select Uninstall. The installer launches, and you can select the components to be removed. Alternatively, you can remove StoreFront by right-clicking Citrix StoreFront and selecting Uninstall.
- To remove a VDA, select Citrix Virtual Delivery Agent <version>, then right-click and select Uninstall. The installer launches and you can select the components to be removed.
- To remove the Universal Print Server, select Citrix Universal Print Server, then right-click and select Uninstall.

From the \x64\XenDesktop Setup directory on the installation media, run the XenDesktopServerSetup.exe command.
- To remove one or more core components, use the /remove and /components options.
- To remove all core components, use the /removeall option.

For command and parameter details, see Install using the command line.

For example, the following command removes Studio.
\x64\XenDesktop Setup\XenDesktopServerSetup.exe /remove /components studio

From the \x64\XenDesktop Setup directory on the installation media, run the XenDesktopVdaSetup.exe command.
- To remove one or more core components, use the /remove and /components options.
- To remove all core components, use the /removeall option.

For command and parameter details, see Install using the command line.

For example, the following command removes the VDA and Receiver.
\x64\XenDesktop Setup\XenDesktopVdaSetup.exe /removeall
You can also remove VDAs using a script in Active Directory; see Install or remove Virtual Delivery Agents using scripts in Active Directory.
This release allows you to grow your deployment at a rate that best suits your organization. You can start with a simple default configuration, which you release to additional user groups at a later time. It is important to think about your deployment in terms of user needs and focus your pilot on the users who will see the most immediate benefit.

If you are not ready to transition your users to virtual desktops, you can start with a Remote PC Access deployment that enables them to access their office PCs and take advantage of Citrix HDX features, and then add traditional virtual desktops and application deployments later.

The Delivery Controller, Studio, Director, License Server, and StoreFront can be installed on the same server or on different servers. For example, to manage your deployment remotely, you can install Studio on a different system than the server where you installed the Controller.

- Controller
- Studio
- Director
- License Server — For further information on licensing, see: Licensing.
- Storefront
- The database — By default, during site creation, Studio creates a database on the server where you install the Controller; this database contains:
  - Site database
  - Configuration logging database
  - Monitoring database

Citrix recommends that you change the location of the configuration logging and monitoring databases after you create the site.

Important: You can choose to use a database on a separate server. If you intend using an external database created manually, not created using Studio, ensure your database administrator uses the following collation setting when creating the database: Latin1_General_100_CI_AS_KS (where Latin1_General varies depending on the country; for example Japanese_100_CI_AS_KS). If this collation setting is not specified during database creation, subsequent creation of the service schemas within the database will fail, and an error similar to "<service>: schema requires a case-insensitive database" appears (where <service> is the name of the service whose schema is being created).

- A domain controller running Active Directory. Active Directory is required. Do not install XenApp, XenDesktop, or the SQL Server database on a domain controller.
  Consider the potential implications of the domain name you select. Avoid domain names that may be used elsewhere. For example, the string ‘client’ is also used to access client drive mapping.

- VMs or physical computers hosting the desktops you want to deliver to your users. You install the Virtual Desktop Agent on these machines to manage communications and broker connections.
- User devices running the appropriate client to enable your users to access desktops.
Components

Mar 28, 2015

If you already have experience with a XenDesktop or XenApp environment, it will be helpful to you to identify the components introduced in this release and learn how they work and communicate with each other. Under the new architecture, XenDesktop and XenApp are unified, including management and delivery components, to give administrators a unified management experience.

This figure shows the key components in a typical deployment.

The core components of this release are:

**Director** — Director is a web-based tool that enables IT support and help desk teams to monitor an environment, troubleshoot issues before they become system-critical, and perform support tasks for end users. You can also view and interact with a user's sessions using Microsoft Remote Assistance.

**Receiver** — Installed on user devices, Citrix Receiver provides users with quick, secure, self-service access to documents, applications, and desktops from any of the user's devices including smartphones, tablets, and PCs. Receiver provides on-demand access to Windows, Web, and Software as a Service (SaaS) applications.

**StoreFront** — StoreFront authenticates users to sites hosting resources and manages stores of desktops and applications that users access.

**Studio** — Studio is the management console that enables you to configure and manage your deployment, eliminating the need for separate management consoles for managing delivery of applications and desktops. Studio provides various
wizards to guide you through the process of setting up your environment, creating your workloads to host applications and desktops, and assigning applications and desktops to users.

**License server** — License server manages your product licenses. You must create at least one license server to store and manage your license files.

**Delivery Controller** — Installed on servers in the data center, the Delivery Controller consists of services that communicate with the hypervisor to distribute applications and desktops, authenticate and manage user access, and broker connections between users and their virtual desktops and applications. The Controller manages the state of the desktops, starting and stopping them based on demand and administrative configuration. In some editions, the Controller allows you to install Profile management to manage user personalization settings in virtualized or physical Windows environments. Each site has one or more Delivery Controllers.

**XenServer** — XenServer is an enterprise-class virtual machine infrastructure solution that creates the foundation for delivering virtual desktops and offers advanced management features. Multiple VMs can run on XenServer, which takes advantage of the advanced virtualization features of the latest virtualization-enabled processors from Intel and AMD. For more information about XenServer, see the XenServer documentation in eDocs.

**Virtual Delivery Agent (VDA)** — Installed on server or workstation operating systems, the VDA enables connections for desktops and apps. For Remote PC Access, install the VDA on the office PC.

**Machine Creation Services (MCS)** — A collection of services that work together to create virtual servers and desktops from a master image on demand, optimizing storage utilization and providing a pristine virtual machine to users every time they log on. Machine Creation Services is fully integrated and administrated in Citrix Studio.

**Windows Server OS machines** — VMs or physical machines based on Windows Server operating system used for delivering applications or hosted shared desktops to users.

**Desktop OS machines** — VMs or physical machines based on Windows Desktop operating system used for delivering personalized desktops to users, or applications from desktop operating systems.

**Remote PC Access** — User devices that are included on a whitelist, enabling users to access resources on their office PCs remotely, from any device running Citrix Receiver.

Additional components provide the following features:

**Secure delivery** — When users connect from outside the corporate firewall, this release can use Citrix NetScaler Gateway (formerly Access Gateway) technology to secure these connections with SSL. NetScaler Gateway or NetScaler VPX virtual appliance is an SSL VPN appliance that is deployed in the demilitarized zone (DMZ) to provide a single secure point of access through the corporate firewall.

**WAN optimization** — In deployments where virtual desktops are delivered to users at remote locations such as branch offices, Citrix CloudBridge (formerly Citrix Branch Repeater or WANScaler) technology can be employed to optimize performance. Repeaters accelerate performance across wide-area networks, so with Repeaters in the network, users in the branch office experience LAN-like performance over the WAN. CloudBridge can prioritize different parts of the user experience so that, for example, the user experience does not degrade in the branch location when a large file or print job is sent over the network. HDX WAN Optimization with CloudBridge provides tokenized compression and data deduplication, dramatically reducing bandwidth requirements and improving performance. For more information, see the Citrix CloudBridge documentation.
Citrix StoreFront is included on the XenDesktop media. When planning your StoreFront deployment, consider the following recommendations. For more information about StoreFront, see the StoreFront documentation in Citrix eDocs.

- Citrix recommends hosting StoreFront on a dedicated instance of IIS. Installing other Web applications on the same IIS instance as StoreFront could have security implications for the overall StoreFront infrastructure.
- In a production environment, Citrix recommends using HTTPS to secure communications between StoreFront and users' devices. To use HTTPS, StoreFront requires that the IIS instance hosting the authentication service and associated stores is configured for HTTPS. In the absence of the appropriate IIS configuration, StoreFront uses HTTP for communications.
- StoreFront servers must reside within the same Microsoft Active Directory forest as the XenDesktop and XenApp servers hosting users' resources. All the StoreFront servers in a group must reside within the same domain. To enable smart card and user certificate authentication, users' accounts must be configured within the Active Directory forest containing the StoreFront servers.
- Consider implementing multiple StoreFront servers to ensure high availability if the primary server hosting StoreFront fails.
- Configure the external load balancer to fail over between the servers to ensure users have uninterrupted access to their applications and desktops.
Remote PC Access

Mar 28, 2015

Remote PC Access allows an end user to log on remotely to the physical Windows PC in his or her office from virtually anywhere. The Citrix XenDesktop Virtual Delivery Agent (VDA) on the office PC enables it to register with the Delivery Controller and manages the HDX connection between the machine and end user client devices. The Citrix Receiver running on the client device provides access to all of the applications and data on the office PC.

A user may have multiple desktops, including more than one physical PC or a combination of physical PCs and virtual desktops.

Remote PC Access supports a self-service model for employee remote access to office PCs. After you set up the whitelist of machines that users are permitted to access, the users can join their office PCs to a XenDesktop site themselves, without administrator intervention. Once joined, users can access their office PCs remotely, from any device running Citrix Receiver.

Consider the following when planning your remote PC deployment:

- The following XenDesktop 7 features are not applicable to remote PC deployments:
  - managing hosts
  - creating master images and virtual machines
  - power managing machines (shutdown and restart)
  - delivering hosted applications
  - Personal vDisks
  - client folder redirection

- Active Directory (AD) considerations:
  - Much of the configuration of a remote PC deployment is done through AD. Before configuring the remote PC deployment site, set up your AD Organizational Units (OUs) and security groups and then create AD user accounts. Use these accounts to specify users for the Delivery Groups you will use to provide Remote PC Access.
  - If you modify AD after a machine has been added to a machine catalog, remote PC automatic administration does not reevaluate that assignment. You can manually assign a machine to a different catalog, if needed.
  - If you move or delete Organizational Unit (OU) entries in AD, the OU entries used for remote PC access can become out of date. Virtual Delivery Agents (VDAs) might no longer be associated with the most appropriate machine catalog or Delivery Group, or with any machine catalog or Delivery Group.

- Machine catalog and Delivery Group considerations:
  - A machine can only be assigned to one machine catalog and one Delivery Group at a time.
  - You can keep all machines in a single remote PC machine catalog, or separate them into multiple machine catalogs.
  - When choosing Machine Accounts for a machine catalog, select the lowest applicable Organizational Unit (OU) to avoid potential conflicts with machines in another machine catalog. For example, in the case of Bank/officers/tellers, select tellers.
  - You can allocate all machines from one remote PC machine catalog through a single Delivery Group, or via multiple Delivery Groups.
  - If you have one group of users who require certain policy settings and another group of users who require different settings, assigning the users to different Delivery Groups enables you to filter the HDX policies according to each Delivery Group.
  - If your IT infrastructure assigns responsibility for servicing users based on geographic location, department, or some...
other category, you can group machines and users accordingly to allow for delegated administration. Ensure that each administrator has permissions for both the relevant machine catalogs and the corresponding Delivery Groups.

- If some of your users have office PCs running Windows 7 or Windows 8 and want to use the latest XenDesktop functionality, but other users have office PCs running Windows XP and therefore cannot, create a separate machine catalog and Delivery Group for the Windows XP systems. When choosing machine accounts for that catalog, select the checkbox Some of the machine are running Windows XP.

- **Deployment considerations:**
  - You can set up a remote PC deployment and then add traditional Virtual Desktop Infrastructure (VDI) desktops or applications later. You can also add remote PC desktops to an existing VDI deployment.
  - When installing the VDA, consider whether to enable Windows Remote Assistance (by specifying `/enable_remote_assistance`). This option allows help desk teams using Director to view and interact with a user's sessions via Remote Assistance.
  - Consider how you will deploy the VDA to each office PC. Citrix recommends using electronic software distribution such as Active Directory (AD) scripts, Microsoft System Center Configuration Manager (SCCM), etc. The installation media contains sample AD scripts.
  - Each office PC must be domain-joined with a wired network connection.
  - Windows 7 Aero is not a requirement on the office PC, but is supported.
  - Remote PC Access can be used on most laptop computers. To improve accessibility and deliver the best connection experience, the laptop power saving options should be configured to those of desktop PC. For example:
    - Disable the Hibernate feature.
    - Disable the Sleep feature.
    - Set the close lid action to Do Nothing.
    - Set the press the power button action to Shut Down.
    - Disable video card energy saving features.
    - Disable network interface card energy saving features.
    - Disable battery saving technologies.
  - Not supported for Remote PC Access devices:
    - Docking and undocking the laptop.
    - KVM switches or other components that can disconnect a session.
    - Hybrid PCs (including All-in-One and NVIDIA Optimus laptops and PCs).
    - Install Citrix Receiver on each client device that remotely accesses the office PC. Receiver is available for a wide range of client devices including desktops, laptops, tablets, and smartphones.
    - Multiple users with remote access to the same office PC see the same icon in Receiver. When any user remotely logs on to the PC, that resource is marked unavailable to other users.

- **Smart card considerations:**
  - Smart cards are supported only for remote access to physical office PCs running Windows 7 or Windows 8; smart cards are not supported for office PCs running Windows XP.
  - The following smart cards were tested with remote PC access:
    - Gemalto .Net 2.0 with the Gemalto .Net mini driver
    - Gemalto IDPrime .NET 510 with Gemalto .Net mini driver
    - Gemalto PIV cards with ActivIdentity ActivClient 6.2
  - For more information about smart cards with this release, see Authenticate securely with smart cards.

- **Licensing considerations:**
  - Remote PC Access sessions consume licenses in the same way as other XenDesktop sessions.
Best practices for users to ensure safe connecting and disconnecting

Keyboard and mouse devices should be attached directly to the PC or laptop, not to the monitor or other components that can be powered down while users may be connecting remotely. This setup ensures that users can log on to their physical PCs and have the input devices in functional state. If input devices are attached to a component like a monitor that can be powered down, users may have issues logging on to their workstation after a Remote PC Access session.

- If input devices are attached to components like monitors, then users should not power down such components.
- If a session is unintentionally disconnected, users can reconnect, which should restore the session.
Example deployments

Two examples of typical XenDesktop deployments are:
- Distributed components configuration
- Multiple site configuration

Distributed components configuration

You can distribute the components of your deployment among a greater number of servers, or provide greater scalability and failover by increasing the number of controllers in your site. You can install the management consoles on separate computers to enable you to manage your deployment remotely. A distributed deployment is necessary for an infrastructure based on remote access through NetScaler Gateway (formerly called Access Gateway).

Figure 1. A distributed components configuration

For more information about Citrix NetScaler Gateway for secure remote access, see the product-specific documentation.

Multiple site configuration

If you have multiple regional sites, for example one in Europe and one in the US, you can use Citrix NetScaler to direct user connections to the most appropriate site and StoreFront to deliver desktops and applications to users.
In the following example, a XenDesktop was created in two data centers. Having two sites globally, rather than just one, minimizes the amount of unnecessary WAN traffic. You can use StoreFront to aggregate resources from multiple XenDesktop sites to provide users with a single point of access with NetScaler. Citrix NetScaler accelerates application performance, load balances servers, increases security, and optimizes the user experience. In this example, two NetScalers are used to provide a high availability configuration. The NetScalers are configured for Global Server Load Balancing and positioned in the DMZ to provide a multi-site, fault-tolerant solution. For more information on highly available multi-site configurations with StoreFront, see Set up highly available multi-site store configurations.

A separate Studio console is required to manage each site; sites cannot be managed as a single entity. You can use Director to support users across sites.

Figure 2. A configuration consisting of multiple regional sites and data centers
Database fault tolerance

Sep 19, 2013
This topic outlines ways in which you can increase the level of fault tolerance in your deployment to make sure that business-critical applications and desktops are always available.

Note: For information about configuring the Virtual Delivery Agent (VDA) to operate in high availability mode, see Ensure desktop and application access if Delivery Controllers fail

All information is stored in the site configuration database; Controllers communicate only with the Database and not with each other. A Controller can be unplugged or turned off without affecting other Controllers in the Site. This means, however, that the site configuration database forms a single point of failure. If the database server fails, existing connections to virtual desktops will continue to function until a user either logs off or disconnects from a virtual desktop; new connections cannot be established if the database server is unavailable.

Citrix recommends that you back up the Database regularly so that you can restore from the backup if the database server fails. In addition, there are several high availability solutions to consider for ensuring automatic failover:

- **SQL Mirroring** — This is the recommended solution. Mirroring the Database makes sure that, should you lose the active database server, the automatic failover process happens in a matter of seconds, so that users are generally unaffected. This method, however, is more expensive than other solutions because full SQL Server licenses are required on each database server; you cannot use SQL Server Express edition for a mirrored environment.
- **Using the hypervisor's high availability features** — With this method, you deploy the database as a virtual machine and use your hypervisor's high availability features. This solution is less expensive than mirroring as it uses your existing Host software and you can also use SQL Express. However, the automatic failover process is slower, as it can take time for a new machine to start for the database, which may interrupt the service to users.
- **SQL Clustering** — Microsoft's SQL clustering technology can be used to automatically allow one server to take over the tasks and responsibilities of another server that has failed. However, setting up this solution is more complicated, and the automatic failover process is typically slower than with alternatives such as SQL Mirroring.
- **AlwaysOn Availability Groups** is an enterprise-level high-availability and disaster recovery solution introduced in SQL Server 2012 to enable you to maximize availability for one or more user databases. AlwaysOn Availability Groups requires that the SQL Server instances reside on Windows Server Failover Clustering (WSFC) nodes. For more information, see AlwaysOn Availability Groups (SQL Server).

The configuration process involves tasks an administrator completes using SQL Server management tools before you create the Citrix Site. The remaining tasks occur when the Citrix administrator runs the Site creation wizard.

A mirror environment requires at least two SQL Server machines (for example, SQL Server A and SQL Server B). SQL Server Express edition cannot be used as either a principal or mirror.

Using Microsoft SQL Server management tools, configure the SQL Server databases:

1. Install the SQL Server software on SQL Server A and SQL Server B.
2. On SQL Server A, create the database intended to be used as the principal (for example, myDatabaseMirror). Make sure that the database uses the full recovery model and not the simple model. (The simple model is configured by default, but prevents the transaction log from being backed up.)
Make sure that the collation sequence ends with _CI_AS_KS (That is, it is case insensitive, accent sensitive, and kanatype sensitive).

Enable a Read-Committed snapshot as described in How to Enable Read-Committed Snapshot in XenDesktop. It is important to enable this before the database is mirrored to avoid errors.

3. On SQL Server A, back up the database to a file and copy it to SQL Server B.
4. On SQL Server B, restore the backup file to that server (SQL Server B).
5. On SQL Server A, start mirroring.

The next step depends on whether the Citrix administrator (that is, the person running the Site creation wizard) also has full database privileges:

- If the Citrix administrator has database privileges (the same person is the database administrator and the Citrix administrator), Studio does everything for you:
  1. The Citrix administrator uses Studio to create a Site, specifying the address of the previously-created SQL Server A database and its name (myDatabaseMirrorForXD).
  2. The database scripts are automatically applied and the principal and mirror databases are set.
- If the Citrix administrator does not have database privileges, the Citrix administrator must get help from a database administrator:
  1. The Citrix administrator uses Studio to create a Site, specifying the address of the previously-created SQL Server and its name (myDatabaseMirrorForXD).
  2. In the Site creation wizard, pressing Generate Script generates a mirror script and a primary script. The Citrix administrator gives those scripts to the database administrator, who applies the scripts (the mirror script should be applied first). The database administrator must tell the Citrix administrator when that task is completed.
  3. Back in Studio, the Citrix administrator can now continue and complete the Create Site wizard. The principal and mirror databases are set.

To verify mirroring after creating the Site, run the PowerShell cmdlet `get-configdbconnection` to make sure that the Failover Partner has been set in the connection string to the mirror.

If you later add, move, or remove a Delivery Controller in a mirrored database environment, see Add, remove, or move Controllers for considerations.
If all Delivery Controllers in a XenDesktop site fail, you can configure the Virtual Delivery Agent (VDA) (for Server OS machines and Desktop OS machines) to operate in high availability mode so that users can continue to access and use their desktops. In high availability mode, the VDA will accept direct ICA connections from users, rather than connections brokered by the controller.

Note: This feature is for use only on the rare occasion that communication with all Controllers fails; it is not an alternative to other high availability solutions, such as configuring database fault tolerance and site failover. Before using this feature, refer to the list of limitations below as these have security implications.

Once in high availability mode (which is enabled for 30 days), the VDA attempts to register with a Controller for up to 30 days, while the user continues to use the desktop in this mode. When the Controller later becomes available, the desktop registers and the user's session continues uninterrupted, but any subsequent connection is brokered by the Controller as normal. If after 30 days the desktop is unable to register with the Controller, the desktop stops listening for connections and is no longer available. This means the administrator has 30 days in which to repair the Controller infrastructure and should not become reliant upon high availability mode.

High availability mode is suitable only for use with dedicated desktops, where the mapping between the user and the VDA is known. You cannot configure high availability mode for use with pooled desktops.

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

To enable high availability mode, you:

1. Set the HighAvailability and HaRegistrarTimeout registry keys.
2. Provide users with an ICA launch file that will enable them to make direct ICA connections. You have to create an ICA file for each user who requires this feature; Citrix does not create or distribute ICA files for this purpose.

Set the registry keys

To configure the VDA so that it operates in high availability mode when necessary, add the following registry key(s). You must do this after the VDA has been installed.

1. Add the following registry entry to HKEY_LOCAL_MACHINE\SOFTWARE\Citrix\VirtualDesktopAgent:
   Name: HighAvailability

   Type: REG_DWORD

   Values:

   1 = enables high availability mode

   0 (zero) = disables high availability mode
2. To change the time period that the VDA tries registering with the Controller before initiating high availability mode, also add the following registry entry:

   Name: HaRegistrarTimeout
   Type: REG_DWORD
   Value: number of seconds. The default is 300 seconds.

3. Restart the virtual desktop.

Prepare an ICA launch file

To establish a direct ICA connection to desktops, provide users with an ICA launch file that they can use should communication with the Controller fail. You must create an ICA launch file for each user who requires this feature; Citrix does not create or distribute ICA files for this purpose. For information on how to create ICA files, see http://support.citrix.com/article/CTX127392.

You will need to tell users when it is appropriate to use this ICA launch file and where they can access it from.

High availability mode limitations

High availability mode is suitable for use only with dedicated desktops; you cannot configure this for use with pooled desktops.

In high availability mode, some features are unavailable. These include:

- User roaming. If a user device is already connected to the desktop, users are unable to connect from a different user device.
- Power management. When the desktop powers up, it attempts to register, fails and, after the timeout, enters high availability mode.
- Delivery Controller-originated policies. Policies originating on the Controller, such as those governing client drive mapping and access to the clipboard, will not function as there is no connection to the Controller. Policies originating from the Domain Controller and Local Group Policy are unaffected. Note that policies from a previous registration persist and are applied, so outdated policies might take affect.
- NetScaler Gateway and Remote Access.

High availability mode persists for up to 30 days only, after which the desktop is no longer available.
Security planning

Sep 29, 2015
This topic describes:

- General security best practices when using this release, and any security-related differences between this release and a conventional computer environment
- Managing user privileges
- Deployment scenarios and their security implications
- Remote PC access

Your organization may need to meet specific security standards to satisfy regulatory requirements. This document does not cover this subject, because such security standards change over time. For up-to-date information on security standards and Citrix products, consult http://www.citrix.com/security/.

Keep all machines in your environment up to date with security patches. One advantage is that you can use thin clients as terminals, which simplifies this task.

Protect all machines in your environment with antivirus software.

Protect all machines in your environment with perimeter firewalls, including at enclave boundaries as appropriate.

If you are migrating a conventional environment to this release, you may need to reposition an existing perimeter firewall or add new perimeter firewalls. For example, suppose there is a perimeter firewall between a conventional client and database server in the data center. When this release is used, that perimeter firewall must instead be placed so that the virtual desktop and user device are on one side of it, and the database servers and Delivery Controllers in the data center are on the other side. You should, therefore, consider creating an enclave within your data center to contain the servers and controllers used by this release. You should also consider having protection between the user device and the virtual desktop.

All machines in your environment should be protected by a personal firewall. When you install a Virtual Delivery Agent (VDA), you can choose to have the ports required for component and feature communication opened automatically if the Windows Firewall Service is detected (even if the firewall is not enabled). You can also choose to configure those firewall ports manually. If you use a different firewall, you must configure the firewall manually.

Note: TCP ports 1494 and 2598 are used for ICA and CGP and are therefore likely to be open at firewalls so that users outside the data center can access them. Citrix recommends that you do not use these ports for anything else, to avoid the possibility of inadvertently leaving administrative interfaces open to attack. Ports 1494 and 2598 are officially registered with the Internet Assigned Number Authority (see http://www.iana.org/).

All network communications should be appropriately secured and encrypted as appropriate to match your security policy. You can secure all communication between Microsoft Windows computers using IPSec; refer to your operating system documentation for details about how to do this. In addition, communication between user devices and desktops is secured through Citrix SecureICA, which is configured by default to 128-bit encryption. You can configure SecureICA when you are creating or updating an assignment; see Secure Delivery Groups.

You should grant users only the capabilities they require. Microsoft Windows privileges continue to be applied to desktops...
in the usual way: configure privileges through User Rights Assignment and group memberships through Group Policy. One advantage of this release is that it is possible to grant a user administrative rights to a desktop without also granting physical control over the computer on which the desktop is stored.

When planning for desktop privileges, note:

- By default, when non-privileged users connect to a desktop, they see the time zone of the system running the desktop instead of the time zone of their own user device. For information on how to allow users to see their local time when using desktops, see Configure time zone settings.
- A user who is an administrator on a desktop has full control over that desktop. If a desktop is a pooled desktop rather than a dedicated desktop, the user must be trusted in respect of all other users of that desktop, including future users. All users of the desktop need to be aware of the potential permanent risk to their data security posed by this situation.
- This consideration does not apply to dedicated desktops, which have only a single user; that user should not be an administrator on any other desktop.
- A user who is an administrator on a desktop can generally install software on that desktop, including potentially malicious software. The user can also potentially monitor or control traffic on any network connected to the desktop.

Your user environment can consist either of user devices that are unmanaged by your organization and completely under the control of the user, or of user devices that are managed and administered by your organization. The security considerations for these two environments are generally different.

**Managed user devices**

Managed user devices are under administrative control; they are either under your own control, or the control of another organization that you trust. You may configure and supply user devices directly to users; alternatively, you may provide terminals on which a single desktop runs in full-screen-only mode. You should follow the general security best practices described above for all managed user devices. This release has the advantage that minimal software is required on a user device.

A managed user device can be set up to be used in full-screen-only mode or in window mode:

- If a user device is configured to be used in full-screen-only mode, users log on to it with the usual Log On To Windows screen. The same user credentials are then used to log on automatically to this release.
- If a user device is configured so that users see their desktop in a window, users first log on to the user device, then log on to this release through a Web site supplied with the release.

**Unmanaged user devices**

User devices that are not managed and administered by a trusted organization cannot be assumed to be under administrative control. For example, you might permit users to obtain and configure their own devices, but users might not follow the general security best practices described above. This release has the advantage that it is possible to deliver desktops securely to unmanaged user devices. These devices should still have basic antivirus protection that will defeat keylogger and similar input attacks.

**Data storage considerations**

When using this release, you can prevent users from storing data on user devices that are under their physical control. However, you must still consider the implications of users storing data on desktops. It is not good practice for users to store data on desktops; data should be held on file servers, database servers, or other repositories where it can be
appropriately protected.

Your desktop environment may consist of various types of desktops, such as pooled and dedicated desktops:

- Users should never store data on desktops that are shared amongst users, such as pooled desktops.
- If users store data on dedicated desktops, that data should be removed if the desktop is later made available to other users.

Mixed-version environments

Mixed-version environments are inevitable during some upgrades. Follow best-practice and minimize the time that Citrix components of different versions co-exist.

In mixed-version environments security policy, for example, may not be uniformly enforced.

Note: This is typical of other software products; the use of an earlier version of Active Directory only partially enforces Group Policy with later versions of Windows.

The following scenario describes a security issue that can occur in a specific mixed-version Citrix environment. When Citrix Receiver 1.7 is used to connect to a virtual desktop running the Virtual Delivery Agent in XenApp and XenDesktop 7.6 Feature Pack 2, the policy “Allow file transfer between desktop and client” is enabled in the Site but cannot be disabled by a Delivery Controller running XenApp and XenDesktop 7.1. It does not recognize the policy, which was released only in the later version of the product. This policy allows users to upload and download files to their virtual desktop – the security issue. To work around this, upgrade the Delivery Controller, or a standalone instance of Studio, to Version 7.6 Feature Pack 2 and then use GP to disable the policy. Alternatively, use local policy on all affected virtual desktops.

Remote PC Access implements the following security features:

- Smart card use is supported.
- When a remote session connects, the office PC’s monitor appears as blank.
- Remote PC access redirects all keyboard and mouse input to the remote session, except CTRL+ALT+DEL and USB-enabled smart cards and biometric devices.
- SmoothRoaming is supported for a single user only.
- When a user has a remote session connected to an office PC, only that user can resume local access of the office PC. To resume local access, the user presses Ctrl-Alt-Del on the local PC and then log in with the same credentials used by the remote session. The user can also resume local access by inserting a smart card or leveraging biometrics, if your system has appropriate third-party Credential Provider integration.

Note: This default behavior can be overridden by enabling Fast User Switching via Group Policy Objects (GPOs) or by editing the registry.

By default, remote PC access supports automatic assignment of multiple users to a VDA. In XenDesktop 5.6 Feature Pack 1, administrators could override this behavior using the RemotePCAccess.ps1 PowerShell script. This release uses a registry entry to allow or prohibit multiple automatic remote PC assignments; this setting applies to the entire site.

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

To restrict automatic assignments to a single user:

1. Set the following registry entry on each controller in the site:
   
   HKEY_LOCAL_MACHINE\Software\Citrix\DesktopServer
Name: AllowMultipleRemotePCAssignments

Type: REG_DWORD

Data: See chart

<table>
<thead>
<tr>
<th>Value</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Disable multiple user assignment.</td>
</tr>
<tr>
<td>1</td>
<td>Enable multiple user assignment. This is the default value.</td>
</tr>
</tbody>
</table>

2. If there are any existing user assignments, remove them using XenDesktop SDK commands in order for the VDA to subsequently be eligible for a single automatic assignment. For more information on using the XenDesktop SDK, see About the XenDesktop SDK.

1. Remove all assigned users from the VDA: $machine.AssociatedUserNames | %{ Remove-BrokerUser-Name $_ -Machine $machine }

2. Remove the VDA from the desktop group: $machine | Remove-BrokerMachine -DesktopGroup $desktopGroup

3. Restart the physical office PC.
Host desktops and applications

Feb 26, 2013

To deliver hosted applications and desktops to your users through Studio, evaluate the following environment characteristics.

- What types of users do you support?
  - Perform well-defined tasks that do not need personalization or offline access, such as call center operators.
  - Perform various tasks, and optionally can install and manage their own applications on virtual machines, such as internal, external contractors, third-party collaborators, and other provisional team members.
  - Work from home or other locations, and need access to specific software or data on their corporate desktops to perform their jobs remotely.
  - Need to access locally installed applications directly on their physical devices and access that same application from their virtual desktop.
  - Require the ability to launch Windows applications without requiring network access on their Windows device.

- What delivery environment do you need?
  - Inexpensive server-based delivery to minimize the cost of delivering applications to a large number of users, while providing a secure, high-definition user experience.
  - Centrally managed applications and desktops that users cannot modify.
  - Adequate resources for users who must consumes a single machine when they log on and modify and save applications and data.
  - Access to centrally managed Windows App-V applications and deliver them to user devices.

- What types of applications do you want to deliver?
  - Any Windows application.
  - Applications that might not work well with other applications or might interact with the operation system, such as the NET framework.
  - Applications running on older operating systems such as Windows XP or Windows Vista, and older architectures, such as 32-Bit or 16-Bit.
  - Applications that are delivered to remote users from an office computer.
  - Specialty or niche applications that are not yet virtualized, such as AutoCAD, or applications that require access to local hardware, such as DVD burners.
  - Microsoft supported App-V applications.

- Where are the applications that users need?
  - Physical or virtual machines in a datacenter.
  - Older operating systems such as Windows XP or Windows Vista.
  - Windows App-V servers.
  - Remote user machines.

Once you determine your user and delivery requirements, see Choose an application and desktop delivery method for details about selecting the method for providing applications and desktops to user devices through Studio.
Choose an application and desktop delivery method

May 08, 2013
The method you choose to provide applications or desktops to users depend on the types of applications and desktops you are hosting, the available system resources, and the types of users, and user experience you want to provide.

Choose between the following application and desktop delivery methods:

- Hosted applications or Desktops on:
  - Server OS machines
  - Desktop OS machines
- Hosted desktops on Remote PC Access
- Hosted applications through:
  - Local App Access — Integrates users' locally-installed applications and hosted applications within a hosted desktop environment.
  - Microsoft Application Virtualization applications (App-V) — Deploys App-V applications from a virtual application server as hosted applications to user devices. Once configured as described in Configure App-V resources, these applications are available to add to application Delivery Groups or desktop and application Delivery Groups. Note the following:
    - All versions prior to App-V 5.0 are not supported.
    - The App-V 5.0 client does not provide offline access to applications.
    - Use the following version combinations for App-V and XenApp or XenDesktop components:

<table>
<thead>
<tr>
<th>App-V version</th>
<th>XenDesktop or XenApp versions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delivery Controller</td>
</tr>
<tr>
<td>5.0 RSTM</td>
<td>XenDesktop 7.0, 7.1, 7.5</td>
</tr>
<tr>
<td></td>
<td>XenApp 7.5</td>
</tr>
<tr>
<td>5.0 Service Pack 1</td>
<td>XenDesktop 7.0, 7.1, 7.5</td>
</tr>
<tr>
<td></td>
<td>XenApp 7.5</td>
</tr>
<tr>
<td>5.0 Service Pack 2</td>
<td>XenDesktop 7.0, 7.1, 7.5</td>
</tr>
<tr>
<td></td>
<td>XenApp 7.5</td>
</tr>
</tbody>
</table>

Regardless of the method you choose, applications and desktops are virtually delivered to a user devices from a machine in a Delivery Group to which the user is assigned.

These sections describe the situations, users, and considerations for selecting Studio application and desktop delivery methods.
<table>
<thead>
<tr>
<th>Use Case</th>
<th>You want</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Inexpensive server-based delivery to minimize the cost of delivering applications to a large number of users, while providing a secure, high-definition user experience.</td>
</tr>
<tr>
<td></td>
<td>Your users</td>
</tr>
<tr>
<td></td>
<td>Perform well-defined tasks and do not require personalization or offline access to applications. Users may include task workers such as call center operators and retail workers, or users that share workstations.</td>
</tr>
<tr>
<td></td>
<td>Application types</td>
</tr>
<tr>
<td></td>
<td>Any application.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Benefits and considerations</th>
<th>Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Manageable and scalable solution within your datacenter.</td>
</tr>
<tr>
<td></td>
<td>Most cost effective application delivery solution.</td>
</tr>
<tr>
<td></td>
<td>Hosted applications are managed centrally and users cannot modify the application, providing a user experience that is consistent, safe, and reliable.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users must be online to access their applications.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>User experience</th>
<th>User requests one or more applications from StoreFront, their Start menu, or a URL you provide to them.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Applications are delivered virtually and display seamlessly in high definition on user devices.</td>
</tr>
<tr>
<td></td>
<td>Depending on profile settings, user changes are saved when the user's application session ends. Otherwise, the changes are deleted.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Process, host, and deliver applications</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Application processing takes place on hosting machines, rather than on the user devices.</td>
</tr>
<tr>
<td></td>
<td>The hosting machine can be a physical or a virtual machine.</td>
</tr>
<tr>
<td></td>
<td>Host</td>
</tr>
<tr>
<td></td>
<td>Applications and desktops reside on a Server OS machine.</td>
</tr>
<tr>
<td></td>
<td>Machines become available through machine catalogs.</td>
</tr>
<tr>
<td></td>
<td>Delivery</td>
</tr>
</tbody>
</table>
Machines within machine catalogs are organized into Delivery Groups that deliver the same set of applications to groups of users.

Server OS machines support:

- Desktop and applications Delivery Groups that host both desktops and applications.
- Application Delivery Groups that host only applications.

<table>
<thead>
<tr>
<th>Session management and assignment</th>
<th>Sessions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server OS machines run multiple sessions from a single machine to deliver multiple applications and desktops to multiple, simultaneously connected users. Each user requires a single session from which they can run all their hosted applications.</td>
<td></td>
</tr>
<tr>
<td>For example, a user logs on and requests an application. One session on that machine becomes unavailable to other users. A second user logs on and requests an application which that machine hosts. A second session on the same machine is now unavailable. If both users request additional applications, no additional sessions are required because a user can run multiple application using the same session. If two more users log on and request desktops, and two sessions are available on that same machine, that single machine is now using four sessions to host four different users.</td>
<td></td>
</tr>
<tr>
<td>Random machine assignments</td>
<td></td>
</tr>
<tr>
<td>Within the Delivery Group to which a user is assigned, a machine on the least loaded server is selected. A machine with session availability is randomly assigned to deliver applications to a user when that user logs on.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Use Case</th>
<th>You want</th>
</tr>
</thead>
<tbody>
<tr>
<td>A client-based application delivery solution that is secure, provides centralized management, and supports a large number of users per host server (or hypervisor), while providing users with applications that display seamlessly in high-definition.</td>
<td></td>
</tr>
<tr>
<td>Your users</td>
<td></td>
</tr>
<tr>
<td>Are internal, external contractors, third-party collaborators, and other provisional team members.</td>
<td></td>
</tr>
<tr>
<td>Your users do not require off line access to hosted applications.</td>
<td></td>
</tr>
<tr>
<td>Application types</td>
<td></td>
</tr>
<tr>
<td>Applications that might not work well with other applications or might interact with the operation system, such as Microsoft .NET framework. These types of applications are ideal for hosting on virtual machines.</td>
<td></td>
</tr>
<tr>
<td>Applications running on older operating systems such as Windows XP or Windows Vista, and older architectures, such as 32-bit or 16-bit. By isolating each application on its own virtual machine, if one machine fails, it does not impact other users.</td>
<td></td>
</tr>
<tr>
<td>Benefits and considerations</td>
<td>Benefits</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td>Applications and desktops on the master image are securely managed, hosted, and run on machines within your datacenter, providing a more cost effective application delivery solution.</td>
</tr>
<tr>
<td></td>
<td>● On log on, users can be randomly assigned to a machine within a Delivery Group that is configured to host the same application.</td>
</tr>
<tr>
<td></td>
<td>● You can also statically assign a single machine to deliver an application to a single user each time that user logs on. Statically assigned machines allow users to install and manage their own applications on the virtual machine.</td>
</tr>
<tr>
<td>Considerations</td>
<td>Considerations</td>
</tr>
<tr>
<td></td>
<td>Running multiple sessions is not supported on Desktop OS machines. Therefore, each user consumes a single machine within a Delivery Group when they log on, and users must be online to access their applications.</td>
</tr>
<tr>
<td></td>
<td>This method may increase the amount of server resources for processing applications and increase the amount of storage for users’ Personal vDisks.</td>
</tr>
</tbody>
</table>

| User experience | The same seamless application experience as hosting shared applications on Server OS machines. |

<table>
<thead>
<tr>
<th>Process, host, and deliver applications</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The same as Server OS machines except they are virtual Desktop OS machines.</td>
</tr>
<tr>
<td>Host</td>
<td>Host</td>
</tr>
<tr>
<td></td>
<td>The same as Server OS machines except they are virtual Desktop OS machines.</td>
</tr>
<tr>
<td>Delivery</td>
<td>Delivery</td>
</tr>
<tr>
<td></td>
<td>The same as Server OS machines except Desktop OS machines can exist only in a desktop Delivery Group.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Session management and assignment</th>
<th>Sessions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Desktop OS machines run a single desktop session from a single machine. When accessing applications only, a single user can use multiple applications (and is not limited to a single application) because the operating system sees each application as a new session.</td>
</tr>
<tr>
<td>Random and static machine assignments</td>
<td>Random and static machine assignments</td>
</tr>
<tr>
<td></td>
<td>Within a Delivery Group to which a user is assigned, when users log on they can access:</td>
</tr>
<tr>
<td>● Statically assigned machine so that each time the user logs on to the same machine.</td>
<td>● Statically assigned machine so that each time the user logs on to the same machine.</td>
</tr>
<tr>
<td>● Randomly assigned machine that is selected based on session availability.</td>
<td>● Randomly assigned machine that is selected based on session availability.</td>
</tr>
</tbody>
</table>
### Use Case

- You want to provide: Employees with secure remote access to a physical computer without using a VPN. For example, the user may be accessing their physical desktop PC from home or through a public Wi-fi hotspot. Depending upon the location, you may want to restrict the ability to print or copy and paste outside of the desktop.

  Bring your own device support without migrating desktop images into the datacenter.

#### Your users

Employees or contractors that have the option to work from home, but need access to specific software or data on their corporate desktops to perform their jobs remotely.

#### Host

The same as Desktop OS machines.

#### Application types

Applications that are delivered from an office computer and display seamlessly in high definition on the remote user's device.

### Benefits and considerations

- Provides a manageable and scalable solution within your datacenter.
- Capitalizes on your existing physical desktop investment.
- You can optionally allow users to start their remote machines through the Microsoft Configuration Manager Wake On LAN feature. Otherwise, their remote machines must be kept online for users to access their applications.

### User experience

User requests a connection to their office machine.

### Process, host, and deliver applications

#### Process

Application processing takes place on the office computer, rather than on the remote user's device.

#### Host

The same as Desktop OS machines.

#### Deliver

The same as Desktop OS machines.

### User management and

Remote PC Access machines run a single connection from a single machine, which provides access to a user's office computer.
Use Case | You want
---|---
To integrate users' locally installed applications and hosted applications within a hosted desktop environment.

Your users
Want to access applications installed locally on their physical laptop, PC, or other device directly from their virtual desktop.

Application types
- Video conferencing software such as GoToMeeting.
- Specialty or niche applications that are not yet virtualized, such as AutoCAD.
- Applications that require access to local hardware, such as DVD burners.

Benefits | Provides a flexible application delivery solution. If users have local applications that you cannot virtualize or that IT does not maintain, those applications still behave as though they are installed on a virtual desktop.

Because applications are installed and run on the user's device, less storage and processing resources are required on the network.

Users can access all their applications from one location. This eliminates the need to toggle between a virtual and physical desktop to use an application.

User experience | Users can start and seamlessly display the application:

- From a shortcut on their virtual desktop.
- Directly on their physical desktop.

Application changes are saved to users' devices.

Process, host, and deliver applications | Process and host
Applications are installed and run locally on the user's device.

Delivery
The user launches the locally-installed application using a shortcut to locally-installed applications on the user's virtual desktops.

Session management | Sessions
When a user disconnects from a virtual desktop session, locally-running applications remain on the
User's local machine in local application windows. User-hosted applications also remain on the user's local machine.

If a user logs off the session or shuts down the virtual desktop, all locally-running application windows in the session are closed, just as with any other local application.

Use Case

You want
To centrally manage Windows App-V applications and deliver them to a user's device using Citrix Receiver.

Your users
Require the ability to launch Windows applications without requiring network access on their Windows device.

Applications types
All Microsoft supported App-V applications.

Benefits and considerations

Benefits
Supported on both Server OS machines and Desktop OS machines.
This is a scalable and affordable solution that shares applications by multiple App-V users from a single machine.

Considerations
All versions prior to App-V 5.0 are not supported.
The App-V 5 client does not support offline access to applications.

User experience
Same as hosted shared applications except that the user always requests the application using Citrix Receiver, then the application is seamlessly delivered to the user's device.

Process, host, and deliver applications

Process and host
Applications are streamed from App-V servers and then run on machines within your datacenter.

Delivery
When a user requests an application using Receiver, that application is delivered to those users from desktop and application Delivery Groups to which users are assigned.

Session

Sessions
| management and assignment | Each user's machine runs a single session from which the user can run all assigned applications. |
Evaluate applications

May 14, 2013
To evaluate and solve compatibility issues for deploying desktop and web applications, or for operating system upgrades, the AppDNA application migration software provides definitive information to help accelerate your deployment.

AppDNA helps you determine the effects of application issues and proposed implementations on your environment, devices, and users.

Because AppDNA integrates with Microsoft System Center Configuration Manager and Active Directory, you can easily import applications that are in ConfigMgr and use their silent switches during any automated repackaging. You can also:

- Create application groups that represent your organization's structure in AppDNA
- View the reports and assessment results according to these groups

AppDNA performs a static analysis of each application's files, registry entries, and API use (that is, the application's "DNA") to assess whether the application is suitable for:

- A variety of platforms, including new versions of Windows servers and clients
- A 64-bit server environment
- A shared server-based deployment
- Multiple simultaneous users in separate sessions
- Application virtualization with App-V

When AppDNA detects potential application issues on a particular platform, it provides detailed information about steps that you can take to resolve the issue.

For full documentation of AppDNA, including system requirements, installation instructions, and getting started information, see the AppDNA in eDocs.
Enhance the user experience for mobile devices

Mar 04, 2013
XenDesktop delivers a superior user experience for mobile users, often with no special configuration required. Integrate XenDesktop with other Citrix products to provide additional features to your mobile workforce.

<table>
<thead>
<tr>
<th>Feature:</th>
<th>Provided by:</th>
</tr>
</thead>
<tbody>
<tr>
<td>High definition user experience on 3G and 4G networks</td>
<td>HDX in XenDesktop&lt;br&gt; In most cases, the default Citrix policy settings provide the best user experience.</td>
</tr>
<tr>
<td>Touch-friendly interface to virtual desktops and applications, optimized for tablet devices</td>
<td>Mobile Experience policy settings in XenDesktop&lt;br&gt; The default settings generally provide the best user experience.</td>
</tr>
<tr>
<td>Remote access to office PCs from other devices, including smart phones, tablets, laptops, and PCs</td>
<td>Remote PC Access in XenDesktop&lt;br&gt; Citrix Receiver</td>
</tr>
<tr>
<td>Automatically provisioned applications to all users of a store</td>
<td>Citrix StoreFront&lt;br&gt; Citrix App Controller</td>
</tr>
<tr>
<td>Email-based account discovery, enabling users to set up an account by entering their email address</td>
<td>Citrix StoreFront&lt;br&gt; Citrix NetScaler Gateway (for remote connections)</td>
</tr>
<tr>
<td>Secure access to stores, desktops and applications</td>
<td>Citrix StoreFront&lt;br&gt; Citrix NetScaler Gateway</td>
</tr>
<tr>
<td>Simplified print management that allows network printing from any device</td>
<td>Universal Print Server in XenDesktop</td>
</tr>
<tr>
<td>Optimized performance and delivery of services to branch offices and mobile users</td>
<td>Citrix CloudBridge</td>
</tr>
<tr>
<td>Role-based management, configuration, and security for corporate and employee-owned mobile devices</td>
<td>Mobile Solutions Bundle (XenMobile MDM and CloudGateway)</td>
</tr>
</tbody>
</table>
Active Directory

Aug 11, 2014

Active Directory is required for authentication and authorization. The Kerberos infrastructure in Active Directory is used to guarantee the authenticity and confidentiality of communications with the Delivery Controllers. For information about Kerberos, see the Microsoft documentation.

You need any of these functional levels for the forest and domain:

- Windows 2000 native
- Windows Server 2003
- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2

To use Policy Modeling, the domain controller must be running on a server whose operating system is Windows Server 2003 to Windows Server 2012 R2; this does not affect the domain functional level.

This product supports:

- Deployments in which the user accounts and computer accounts exist in domains in a single Active Directory forest. User and computer accounts can exist in arbitrary domains within a single forest. All domain functional levels and forest functional levels are supported in this type of deployment.

- Deployments in which user accounts exist in an Active Directory forest that is different from the Active Directory forest containing the computer accounts of the controllers and virtual desktops. In this type of deployment, the domains containing the Controller and virtual desktop computer accounts must trust the domains containing user accounts. Forest trusts or external trusts can be used. All domain functional levels and forest functional levels are supported in this type of deployment.

- Deployments in which the computer accounts for Controllers exist in an Active Directory forest that is different from one or more additional Active Directory forests that contain the computer accounts of the virtual desktops. In this type of deployment, a bi-directional trust must exist between the domains containing the Controller computer accounts and all domains containing the virtual desktop computer accounts. In this type of deployment, all domains containing Controller or virtual desktop computer accounts must be at "Windows 2000 native" functional level or higher. All forest functional levels are supported.

- Writable domain controllers. Read-only domain controllers are not supported.

Optionally, Virtual Delivery Agents (VDAs) can use information published in Active Directory to determine which Controllers they can register with (discovery). This method is supported primarily for backward compatibility, and is available only if the VDAs are in the same Active Directory forest as the Controllers. For information about this discovery method see Active Directory OU-based Controller discovery and CTX118976.
Deploy in a multiple forest Active Directory environment

Sep 02, 2015
Note: This information applies to minimum version XenDesktop 7.1 and XenApp 7.5. It does not apply to earlier versions of XenDesktop or XenApp.

In an Active Directory environment with multiple forests, if one-way or two-way trusts are in place you can use DNS forwarders for name lookup and registration. To allow the appropriate Active Directory users to create computer accounts, use the Delegation of Control wizard. Refer to Microsoft documentation for more information about this wizard.

No reverse DNS zones are necessary in DNS infrastructure if appropriate DNS forwarders are in place between forests.

In this release, the SupportMultipleForest key is necessary if the Virtual Delivery Agent (VDA) and Delivery Controller are in separate forests, regardless of whether the Active Directory and NetBios names are different. The SupportMultipleForest key is only necessary on the VDA. Use the following information to add the registry key:

- **HKEY_LOCAL_MACHINE\Software\Citrix\VirtualDesktopAgent\**
  - Name: SupportMultipleForest
  - Type: REG_DWORD
  - Data: 0x00000001 (1)

Set the value to 1.

You might need reverse DNS configuration if your DNS namespace is different than that of Active Directory.

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

If external trusts are in place during setup, the ListOfSIDs registry key is required. The ListOfSIDs registry key is also necessary if the Active Directory FQDN is different than the DNS FQDN or if the domain containing the Domain Controller has a different Netbios name than the Active Directory FQDN. To add the registry key, use the following information:

- For a 32-bit or 64-bit VDA, locate the registry key: **HKEY_LOCAL_MACHINE\Software\Citrix\VirtualDesktopAgent\**
  - Name: ListOfSIDs
  - Type: REG_SZ
  - Data: Security Identifier (SID) of the Controllers

To obtain your domain SID, use ADExplorer or XDPing.

When external trusts are in place, you will also need to make the following changes on the VDA:

1. Locate the file *<ProgramFiles>\Citrix\Virtual Desktop Agent\brokeragent.exe.config*
2. Make a backup copy of the file
3. Open the file in a text editing program such as Notepad
4. Locate the text `allowNtlm="false"` and change the text to `allowNtlm="true"`
5. Save the file

After the ListOfSIDs registry key has been added and the brokeragent.exe.config file has been edited, you will need to restart the Citrix Desktop Service so that the changes will be applied.

The following table indicates which trust types are supported:
<table>
<thead>
<tr>
<th>Trust type</th>
<th>Transitivity</th>
<th>Direction</th>
<th>Supported in this release</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent and child</td>
<td>Transitive</td>
<td>Two-way</td>
<td>Yes</td>
</tr>
<tr>
<td>Tree-root</td>
<td>Transitive</td>
<td>Two-way</td>
<td>Yes</td>
</tr>
<tr>
<td>External</td>
<td>Nontransitive</td>
<td>One-way or two-way</td>
<td>Yes</td>
</tr>
<tr>
<td>Forest</td>
<td>Transitive</td>
<td>One-way or two-way</td>
<td>Yes</td>
</tr>
<tr>
<td>Shortcut</td>
<td>Transitive</td>
<td>One-way or two-way</td>
<td>Yes</td>
</tr>
<tr>
<td>Realm</td>
<td>Transitive or nontransitive</td>
<td>One-way or two-way</td>
<td>No</td>
</tr>
</tbody>
</table>

For more information about complex Active Directory environments, see [CTX134971](https://docs.citrix.com).
You can transfer data and settings from a XenDesktop 4 farm to a XenDesktop 7.x Site using the Migration Tool, which includes the following components:

- The Export Tool, XdExport, to export XenDesktop 4 farm data.
- An XML editor to review and edit the XML file, whose default name is XdSettings.xml.
- The Import Tool, XdImport, that imports the data by running the PowerShell script Import-XdSettings.ps1.

To successfully use the Migration Tool, both deployments must have the same:

- Hypervisor version (for example, XenServer 6.2)
- Active Directory environment

**XenApp Restrictions**

- You cannot migrate earlier XenApp versions to the current version
- You cannot migrate XenApp farms to XenDesktop
- You cannot migrate XenDesktop 4 to XenApp

**Prerequisites**

Perform the following tasks before migration:

- Make sure that you understand which data can be exported and imported, and how this applies to your own deployment. Information on which types of data are exported and imported is available at Data import and export details.
- Citrix strongly recommends that you manually back up the Site Database, so that you can restore it if any issues are discovered.
- Set up a XenDesktop 7.x Site, including its database. For full details of database requirements, see System requirements for XenDesktop 7.1.
- To migrate from XenDesktop 4 to XenDesktop 7.x, all VDAs must be at a XenDesktop 5.x level so that they are compatible with both XenDesktop 4 and XenDesktop 7.x controllers. Once the controller infrastructure is fully running XenDesktop 7.x, Windows 7 VDAs can be upgraded to XenDesktop 7.x. For specific information see Migration examples.

The following figure summarizes the migration process.
Migration steps

1. In the Desktop Studio console on the XenDesktop 4 Controller, put all machines you want to export into maintenance mode.
2. Export data and settings from your XenDesktop 4 farm to an XML file using the export tool. To do this, see Export from a XenDesktop 4 farm.
3. Edit the XML file so that it contains only the data and settings you want to import into your XenDesktop 7.x Site. For further details of how to edit the XML file, see Edit the Migration Tool XML file.
4. Import the data and settings from the XML file to your XenDesktop 7.x Site. To use the import tool to complete this step, see Import XenDesktop 4 data.
5. To make additional changes, repeat steps 4 and 5. After making changes, you may want to import additional desktops into existing desktop groups. To do so, use the Mergedesktops option as described in Import XenDesktop 4 data.
6. Complete the post-migration tasks described in Post-migration tasks.

For migration examples, see Migration examples.

New concepts and features

Compared to XenDesktop 4, this release has many new features and introduces new concepts. For new concepts, terminology, and system requirements, see About XenDesktop.
Data import and export details

Apr 08, 2013
To migrate XenDesktop 4, perform these general processes:

- Export data from XenDesktop 4 and import into this release.
- Export and import user policy settings.

Migration limitations
Not all data and settings are exported. The following configuration items are not migrated because they are exported but not imported:

- Administrators
- Delegated Administration settings
- Desktop Group folders
- Licensing configuration
- Registry keys

See XenDesktop 4 components that are not migrated for information about other components that are not migrated.

These use cases are not directly supported in migration:

- Merging settings of policies or Desktop Group or hosting settings.
- Merging private desktops into random Delivery Groups.
- Adjusting existing component settings through the migration tools.

Migration process summary
Although not all inclusive, this table describes what happens to the most significant data during migration to this release.

<table>
<thead>
<tr>
<th>Data type</th>
<th>Exported?</th>
<th>Imported?</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desktop Groups</td>
<td>Y</td>
<td>Y</td>
<td>Desktop Groups become Delivery Groups in this release.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Desktop Group icons are not exported.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SecureICARequired is set to True if the DefaultEncryptionLevel in</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>XenDesktop 4 is not Basic.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If a Desktop Group in the XenDesktop 4 farm has the same name as a</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Delivery Group in the XenDesktop 7.x Site, you can add desktops</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>belonging to the XenDesktop 4 group to a Delivery group of the same</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>name in the target Site.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>To do this, you must specify the MergeDesktops parameter when you run</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>the import tool. The settings of the XenDesktop 7.x Delivery Group</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>are not overwritten with the settings of the XenDesktop 4 group. If</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>this parameter is not specified and there is a group with the same</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>name as one defined in the XML file, the tool displays an error and</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>stops before any</td>
</tr>
<tr>
<td>Data type</td>
<td>Exported?</td>
<td>Imported?</td>
<td>Notes</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Desktops</td>
<td>Y</td>
<td>Y</td>
<td>You cannot add private desktops to a random Delivery Group. Random desktops cannot be added to a static Delivery Group.</td>
</tr>
</tbody>
</table>
| Machines           | Y         | Y         | Machines are imported into four machine catalogs. The following machine catalogs are automatically created in the XenDesktop 7.x Site by the import tool:  
  - Imported existing random (for pooled VMs)  
  - Imported existing static (for assigned VMs)  
  - Imported physical random (for pooled PCs or blades)  
  - Imported physical static (for private PCs or blades). Any subsequent import of machines uses the same four machine catalogs. |
| Pool management    | Y         | Y         | Includes multi-pool pools, and idle pool settings including schedule.  
  PeakBufferSizePercent is set to 10% by default.  
  OffPeakBufferSizePercent is set to 10% by default.  
  Any unselected days in the Business days setting on XenDesktop 4 are imported as part of the Weekend power time scheme in this release.  
  HostingXD4 action times are rounded up to the nearest minute.  
  Start times are rounded down to the nearest hour.  
  End times are rounded up to the nearest hour. |
| Farm settings      | Y         | Y         | The following farm settings are imported as a Machine policy:  
  - IcaKeepAlive  
  - AutoClientReconnect  
  - SessionReliability  
  The setting to enable Flash player is not imported. |
| Policies           | Y         | Y         | Some policy data is imported. Filters, settings, and printers are imported as User policies. For further details of user policy export and import, see the other table in this topic.  
  - New access policy rules are created from XenDesktop 4 group settings.  
  - When policies are imported, their relative priority order is preserved. However, they are always added with a higher priority than any existing policies on the XenDesktop 7.x Site.  
  - Policy merging is not supported. |
There is no option to import policies into Active Directory. They are always stored in the Site.

<table>
<thead>
<tr>
<th>Data type</th>
<th>Exported?</th>
<th>Imported?</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>User assignments</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Hypervisor settings</td>
<td>Y</td>
<td>Y</td>
<td>This parameter is required. Hypervisor addresses are exported, but not the credentials required to access those hypervisors. To create hypervisor connections in the XenDesktop 7.x Site, extract the addresses from the XML file and create a Powershell hash table that maps them to the relevant credential instances. You then specify this hash table in the import tool HypervisorConnectionCredentials parameter. For further details about how to create the table, see Import XenDesktop 4 data. Merging or updating hypervisor settings for existing Desktop Groups and hypervisor connections is not supported.</td>
</tr>
<tr>
<td>Administrators</td>
<td>Y</td>
<td>N</td>
<td>No administrator data is imported, including data about delegated administrators. You create new administrators for your XenDesktop 7.x Site.</td>
</tr>
<tr>
<td>Licensing configuration</td>
<td>Y</td>
<td>N</td>
<td>Includes information such as the License Server name and edition. License files are not exported.</td>
</tr>
<tr>
<td>Desktop Group folders</td>
<td>Y</td>
<td>N</td>
<td>This release does not support Desktop Group folders. If there are duplicate Desktop Group names (because different folders in the XenDesktop 4 farm contained groups with the same names), if you do not edit names in the XML file, the Import Tool halts.</td>
</tr>
<tr>
<td>Registry keys</td>
<td>Y</td>
<td>N</td>
<td>For information on implementing registry keys, see Post-migration tasks.</td>
</tr>
</tbody>
</table>

User policy data

This table describes how User policy data is exported and imported. For detailed information about policies, see Policy settings reference.

<table>
<thead>
<tr>
<th>XenDesktop 4 category and setting</th>
<th>XML file</th>
<th>XenDesktop 7.x category and setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bandwidth\Visual Effects\Session Limits</td>
<td>ClientOEMVCBandwidth</td>
<td>Not imported</td>
</tr>
<tr>
<td>XenDesktop 4 category and setting</td>
<td>XML file</td>
<td>XenDesktop 7.x category and setting</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Client Devices\Resources\Other</td>
<td>DisableOEMVirtualChannels</td>
<td>Not imported</td>
</tr>
<tr>
<td>Turn off OEM virtual channels</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User Workspace\Time Zones</td>
<td>DoNotUseClientLocalTime</td>
<td>Not imported</td>
</tr>
<tr>
<td>Do not use client's local time</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Security\Encryption</td>
<td>ClientSecurityRequirement</td>
<td>Not imported</td>
</tr>
<tr>
<td>SecureICA encryption</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth\SpeedScreen</td>
<td>LossyCompression settings</td>
<td>ICA\Visual Display\Still Images</td>
</tr>
<tr>
<td>Image acceleration using lossy compression</td>
<td></td>
<td>Lossy compression level</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Lossy compression threshold value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Heavyweight compression</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICA\Visual Display\Moving Images</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Progressive compression level</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Progressive compression threshold value</td>
</tr>
<tr>
<td>Bandwidth\Visual Effects</td>
<td>TurnOffWallpaper</td>
<td>ICA\Desktop UI</td>
</tr>
<tr>
<td>Turn off desktop wallpaper</td>
<td></td>
<td>Desktop wallpaper</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth\Visual Effects</td>
<td>TurnOffMenuWindowAnimation</td>
<td>ICA\Desktop UI</td>
</tr>
<tr>
<td>Menu animation</td>
<td></td>
<td>Menu animation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth\Visual Effects</td>
<td>DoNotShowWindowContentsWhileDragging</td>
<td>ICA\Desktop UI</td>
</tr>
<tr>
<td>Turn off window contents while dragging</td>
<td></td>
<td>View window contents while dragging</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth\Visual Effects\Session</td>
<td>ClientAudioBandwidth__AllowedBandWidth</td>
<td>ICA\Bandwidth</td>
</tr>
<tr>
<td>Limits</td>
<td>XML file</td>
<td>XenDesktop 7.x category and setting</td>
</tr>
<tr>
<td>--------</td>
<td>----------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>Audio</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth\Visual Effects\Session Limits</td>
<td>ClientClipboardBandwidth__AllowedBandWidth</td>
<td>ICA\Bandwidth Clipboard redirection bandwidth limit</td>
</tr>
<tr>
<td>Clipboard</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth\Visual Effects\Session Limits</td>
<td>ClientComBandwidth__AllowedBandWidth</td>
<td>COM port redirection is deprecated in XenDesktop 7.x</td>
</tr>
<tr>
<td>COM Ports</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth\Visual Effects\Session Limits</td>
<td>ClientDriveBandwidth__AllowedBandWidth</td>
<td>ICA\Bandwidth File redirection bandwidth limit</td>
</tr>
<tr>
<td>Drives</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth\Visual Effects\Session Limits</td>
<td>ClientLptBandwidth__AllowedBandWidth</td>
<td>LPT port redirection is deprecated in XenDesktop 7.x</td>
</tr>
<tr>
<td>LPT Ports</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth\Visual Effects\Session Limits</td>
<td>OverallBandwidth__AllowedBandWidth</td>
<td>ICA\Bandwidth Overall session bandwidth limit</td>
</tr>
<tr>
<td>Overall Session</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth\Visual Effects\Session Limits</td>
<td>LimitPrinterBandwidth__AllowedBandWidth</td>
<td>ICA\Bandwidth Printer redirection bandwidth limit</td>
</tr>
<tr>
<td>Printer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Client Devices\Resources\Audio Microphones</td>
<td>ClientAudioMicrophone__TurnOn</td>
<td>ICA\Audio Client microphone redirection</td>
</tr>
<tr>
<td>Sound Quality</td>
<td>ClientAudioQuality__Quality</td>
<td>ICA\Audio Audio quality</td>
</tr>
<tr>
<td>Turn off speakers</td>
<td>DisableClientAudioMapping</td>
<td>ICA\Audio Client audio redirection</td>
</tr>
<tr>
<td>XenDesktop 4 category and setting</td>
<td>XML file</td>
<td>XenDesktop 7.x category and setting</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>Connection</td>
<td>ConnectClientDriveAtLogon__TurnOn</td>
<td>Auto connect drives</td>
</tr>
<tr>
<td>Client Devices\Resources\Drives</td>
<td>DisableClientDriveMapping__DisableFloppyDrive</td>
<td>ICA\File Redirection Floppy drives</td>
</tr>
<tr>
<td>Turn off Floppy disk drives</td>
<td>DisableClientDriveMapping__DisableHardDrive</td>
<td>ICA\File Redirection Fixed drives</td>
</tr>
<tr>
<td>Client Devices\Resources\Drives</td>
<td>DisableClientDriveMapping__DisableCdrom</td>
<td>ICA\File Redirection Optical drives</td>
</tr>
<tr>
<td>Turn off CD-ROM drives</td>
<td>DisableClientDriveMapping__DisableRemote</td>
<td>ICA\File Redirection Network drives</td>
</tr>
<tr>
<td>Client Devices\Resources\Drives</td>
<td>DisableClientDriveMapping__DisableUSB</td>
<td>ICA\File Redirection Removable drives</td>
</tr>
<tr>
<td>Turn off Remote drives</td>
<td>CDMAasyncWrites</td>
<td>ICA\File Redirection User async writes</td>
</tr>
<tr>
<td>Client Devices\Resources\Drives</td>
<td>DisableClientClipboadMapping</td>
<td>ICA Client clipboard redirection</td>
</tr>
<tr>
<td>Turn off clipboard mapping</td>
<td>DisableClientCOMPortMapping</td>
<td>COM port redirection is deprecated in XenDesktop 7.x</td>
</tr>
<tr>
<td>Client Devices\Resources\Ports</td>
<td>DisableClientLPTPortMapping</td>
<td>LPT port redirection is deprecated in XenDesktop 7.x</td>
</tr>
<tr>
<td>Turn off COM ports</td>
<td>RemoteUSBDevices__DisableRemoteUSBDevices</td>
<td>ICA\USB Devices Client USB device redirection</td>
</tr>
<tr>
<td>XenDesktop 4 category and setting</td>
<td>XML file</td>
<td>XenDesktop 7.x category and setting</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Auto-creation</td>
<td></td>
<td>Auto-create client printers</td>
</tr>
<tr>
<td>Printing\Client Printers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Legacy client printers</td>
<td>LegacyClientPrinters__TurnOn</td>
<td>ICA\Printing\Client Printers</td>
</tr>
<tr>
<td>Printer properties retention</td>
<td>ModifiedPrinterProperties__WriteMethod</td>
<td>ICA\Printing\Client Printers</td>
</tr>
<tr>
<td>Print job routing</td>
<td>ClientPrintingForNetworkPrinter__TurnOn</td>
<td>ICA\Printing\Client Printers</td>
</tr>
<tr>
<td>Turn off client printer mapping</td>
<td>DisableClientPrinterMapping</td>
<td>ICA\Printing</td>
</tr>
<tr>
<td>Native printer driver auto-install</td>
<td>PrintDriverAutoInstall__TurnOn</td>
<td>ICA\Printing\Drivers</td>
</tr>
<tr>
<td>Universal driver</td>
<td>ClientPrintDriverToUse</td>
<td>ICA\Printing\Drivers</td>
</tr>
<tr>
<td>Session printers</td>
<td>NetworkPrinters</td>
<td>ICA\Printing</td>
</tr>
<tr>
<td>Choose client’s default printer</td>
<td>DefaultToMainClientPrinter__NetworkDefault</td>
<td>ICA\Printing</td>
</tr>
<tr>
<td></td>
<td>DefaultToMainClientPrinter__TurnOn</td>
<td>ICA\Printing</td>
</tr>
</tbody>
</table>

Legacy client printers and client printer names are configurable settings managed by XenDesktop 4 and XenDesktop 7.x categories and settings.
Export from a XenDesktop 4 farm

Feb 13, 2013

The export tool, XdExport, extracts data from a single XenDesktop 4 farm and produces an XML file from representations of the data values.

The data types exported are described in Data import and export details.

The schema of the XML file resides in the file XdFarm.xsd, which is included in the migration tool download: XdExport.zip and XdImport.zip.

To run the export tool

Run the tool on a machine that is a XenDesktop 4 Controller in the farm from which you want to export data. This machine must have the XenDesktop 4 PowerShell SDK installed.

You must have the following permissions to export the data:

- The user identity of at least read-only Citrix administrator of the farm.
- Permission to read the registry.

Although not recommended, you can run the tool while the XenDesktop Controller is in active use (for example, users are logged in to VDAs).

Citrix strongly recommends:

- The XenDesktop 4 Controller on which you run the tool be up-to-date with public hotfixes.
- Not making configuration changes to the Site while the export is running (for example, removing Desktop Groups).

1. Download XdExport.zip and extract the files to the XenDesktop 4 Controller.
2. At a command line prompt, run XdExport.exe. You can include parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Verbose</td>
<td>Generates messages providing detailed progress information.</td>
</tr>
<tr>
<td>-FilePath &lt;path&gt;</td>
<td>Indicates the location of the XML file to which the farm data is exported.</td>
</tr>
<tr>
<td></td>
<td>Default = \XdSettings.xml</td>
</tr>
<tr>
<td>-Overwrite</td>
<td>Overwrites any file existing in the location specified in -FilePath.</td>
</tr>
<tr>
<td></td>
<td>If you do not supply this parameter and an output file already exists, the tool fails with the following error message: Error: File already exists. Specify -Overwrite to allow the file to be overwritten.</td>
</tr>
<tr>
<td>-? or -help</td>
<td>Displays text describing the parameters and exits without exporting any data.</td>
</tr>
</tbody>
</table>

3. If the tool runs successfully, the message Done appears. The XdSettings.xml file resides in the location specified in the
FilePath parameter.

If the tool fails, an error message appears.

When you have successfully run the Export tool, review and edit the XML file as described in Edit the Migration Tool XML file.
Edit the Migration Tool XML file

Feb 25, 2013

Before importing data to a XenDesktop 7.x Site, check and edit the contents of the XML file generated by the export tool, XdExport, particularly if you migrate in multiple stages and import some users, Delivery Groups, and policies before importing others.

Use any text editor to view or change the file contents, or you can use a specialized XML editor such as Microsoft XML Notepad.

Some elements within the XML content must be present for the XML file to be accepted by the import tool, XdImport.

The required XML schema is defined in the XdFarmxsd file that is supplied as part of the Migration Tool download. Keep the following in mind when working with this file:

- In the file, a minOccurs attribute with a value of 1 or more indicates that particular elements must be present if the parent element is present.
- If the XML file supplied to the Import tool is not valid, the tool halts and an error message appears that should enable you to locate where the problem lies in the XML file.

Import a subset of desktops or Delivery Groups

To import only a subset of Delivery Groups and desktops, edit the contents of the DesktopGroups element. The DesktopGroups element can hold many DesktopGroup elements, and within each DesktopGroup element there is a Desktops element that can contain many Desktop elements.

You must not delete the DesktopGroups element, although you can delete all the DesktopGroup elements and leave it empty. Similarly, within each DesktopGroup element the Desktops element must be present but can be empty of Desktop elements.

Delete Desktop or DesktopGroup elements to avoid importing particular single machines or entire Delivery Groups. For example, the XML file contains:

```
<DesktopGroups>
  <DesktopGroup name="Group1">
    ...
    <Desktops>
      <Desktop sameName="DOMAIN\MACHINE1$">
        ...
      </Desktop>
    </Desktops>
    ...
  </DesktopGroup>
  <DesktopGroup name="Group2">
    ...
    <Desktops>
      <Desktop sameName="DOMAIN\MACHINE2$">
        ...
      </Desktop>
    </Desktops>
    ...
  </DesktopGroup>
```

In this example, your edits prevent Group1 group from being imported. Only Machine3 from the Group2 group would be imported:

```xml
<DesktopGroups>
<DesktopGroup name="Group2">
  <Desktops>
    <Desktop samName="DOMAIN\MACHINE3$">
    ...
    </Desktop>
  </Desktops>
  ...
</DesktopGroup>
</DesktopGroups>
```

Manage Delivery Groups with duplicate names

In XenDesktop 4, Desktop Groups can be organized in folders, Desktop Groups with the same name can appear in different folders, and the internal desktop group name is the name that appears to users.

In this release, Delivery Groups cannot be placed in folders, and each Delivery Group must have a unique internal name, and the name that appears to users can be different from the internal name. To accommodate these differences, you might have to rename Desktop Groups.

For example, in your XenDesktop 4 farm, you could have two different Desktop Groups that appear with the name "My Desktop" to two different users, and you could use Desktop Groups folders to achieve this. If these Delivery Groups are to remain separate in the XenDesktop 7.x Site, you must edit the Desktop Group names in the XML file to make them unique.

If a Delivery Group in the XenDesktop 7.x Site has the same name as a Desktop Group to be imported, and the Delivery Groups are to remain separate in the XenDesktop 7.x Site, you must edit the XenDesktop 4 Desktop Group name in the XML file to keep the name unique in the Site. If the Desktop Group to be imported is really the same as the XenDesktop 7.x Delivery Group, and the machines in the XML file are to be merged into the existing Desktop Group, you do not need to rename the Desktop Group; instead, specify the -MergeDesktops parameter to the Import tool. For example, if the XML file contained:

```xml
<DesktopGroups>
<DesktopGroup name="My Desktop">
  ...
  <Folder>Sales</Folder>
</DesktopGroup>
<DesktopGroup name="My Desktop">
  ...
  <Folder>Finance</Folder>
</DesktopGroup>
</DesktopGroups>
```

Remove the duplicate names as follows:
<DesktopGroups>
<DesktopGroup name="Sales Desktops">
  ...
  <Folder>Sales</Folder>
</DesktopGroup>
<DesktopGroup name="Finance Desktops">
  ...
  <Folder>Finance</Folder>
</DesktopGroup>
</DesktopGroups>

Manage policy imports

You can delete policies from the XML file, and you can specify unique names to avoid policy name duplication. There is no support for merging policies.

Note the following about policy imports:

- When you import policy data, either all policies are imported successfully or, if there is any failure, no policy data is imported.
- Importing large numbers of policies with many settings can take several hours.
- If you import policies in batches, their original prioritization may be affected. When you import policies, the relative priorities of the imported policies are maintained, but they are given higher priority than policies already in the Site. For example, if you have four policies to import with priority numbers 1 to 4, and you decide to import them in two batches, you should import policies with priorities 3 and 4 first, because the second batch of policies automatically gets higher priority.

To import only a subset of policies into the XenDesktop 7.x Site, edit the contents of the Policies element. The Policies element can hold many Policy elements. You must not delete the Policies element, although you can delete all the Policy elements and leave it empty. Delete entire Policy elements to avoid importing particular XenDesktop 4 farm policies. For example, if the XML file contained:

```xml
<Policies>
  <Policy name="Sales Policy">
    ...
  </Policy>
  ...
</Policies>
```

To avoid importing any XenDesktop 4 policies, and avoid clashes with policies already configured in the XenDesktop 7.x Site, edit the file to remove the individual Policy elements as follows:

```xml
<Policies>
</Policies>
```

Alternatively, edit the file so that the policy is imported with a different name as follows:

```xml
<Policies>
  <Policy name="XD4 Sales Policy">
    ...
  </Policy>
  ...
</Policies>
```
Import XenDesktop 4 data

Feb 25, 2013

The import tool, XdImport, reads settings from XenDesktop 4 that are contained in the XML file produced by the export tool, XdExport, and applies those settings to an existing XenDesktop 7.x Site. The Import tool is the PowerShell script Import-XdSettings.ps1.

Not all the data is imported from the XdSettings.xml file; for details of which types of data are imported, see Data import and export details.

To apply only a subset of the exported data, edit the XML file before running the Import tool. For example, you might want to remove desktop groups and policies that are not needed in your XenDesktop 7.x deployment.

The import tool runs successfully if you leave entire elements empty. For example, you can delete all the desktop groups without causing any issues. The tool always validates the XML file before attempting to import any data. For further details of how to edit the XML file, see Edit the Migration Tool XML file.

Data import and export details describes how the data in the XML file migrates to the XenDesktop 7.x Site.

Run XdImport on any machine on which all the XenDesktop 7.x SDKs are installed. You must have the full XenDesktop Administrator identity to run the tool.

Before you import, make sure that you have set up a XenDesktop 7.x Site, including its database. For full details of database requirements, see the Database section in System requirements for XenDesktop 7.1.

Citrix recommends that you complete the import to XenDesktop 7.x before any user testing or general Site configuration occurs. Merge configurations only when the Site is not in use.

To import XenDesktop 4 data

1. Create a XenDesktop 7.x site.
2. Download XdImport.zip and extract the files to the machine on which you plan to run the tool.
3. In a PowerShell session, run Import-XdSettings.ps1. You can specify the following parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>- HypervisorConnectionCredentials</td>
<td>This parameter is required.</td>
</tr>
<tr>
<td></td>
<td>A PowerShell hash table that maps Hypervisor addresses to PSCredential</td>
</tr>
<tr>
<td></td>
<td>instances as required for the creation of Hypervisor connections.</td>
</tr>
<tr>
<td></td>
<td>Default = @{}</td>
</tr>
<tr>
<td></td>
<td>Enter credentials for the Hypervisor to which the XenDesktop 4 farm connects.</td>
</tr>
<tr>
<td></td>
<td>For a single Hypervisor, create the argument as follows:</td>
</tr>
<tr>
<td></td>
<td>$credential = Get-Credential</td>
</tr>
<tr>
<td></td>
<td>$mappings = @&quot;http://&lt;HypervisorIP&gt;&quot; = $credential</td>
</tr>
<tr>
<td></td>
<td>\Import-XdSettings.ps1</td>
</tr>
</tbody>
</table>
Note that the address specified in the hash table must exactly match the address in the XML file.

For example, with both a XenServer and a VMware Hypervisor, create the following argument:

```powershell
$Xencredential = Get-Credential
$VMWcredential = Get-Credential
$mappings = @{"http://<XenHypervisorIP>" = $Xencredential;"http://<VmWHypervisorIP>/SDK" = $VMWcredential}
.Import-XdSettings.ps1
-FilePath.\XdSettings.xml
-HypervisorConnectionCredentials $mappings
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-FilePath &lt;path&gt;</td>
<td>The location of the XML file from which the farm data is to be imported. The value for &lt;path&gt; is required.</td>
</tr>
<tr>
<td>-AdminAddress</td>
<td>The name of a controller in the XenDesktop 7.x Site. Default = localhost</td>
</tr>
<tr>
<td>-MergeDesktops</td>
<td>Adds desktops defined in the XML file to Delivery Groups in the XenDesktop 7.x Site that have the same name as the groups described in the XML file. The associated machines and users are also added. If this parameter is not supplied, no content is added to existing desktop groups in the XenDesktop 7.x Site.</td>
</tr>
<tr>
<td>-SkipMachinePolicy</td>
<td>The script does not create a machine policy that contains site-level settings. If you do not supply this parameter and the machine policy for the site exists, the script fails.</td>
</tr>
<tr>
<td>-WhatIf</td>
<td>Completes a trial run to determine what would be changed in or added to the XenDesktop 7.x Site. Including this parameter sends the information to the log file, but does not change the Site.</td>
</tr>
<tr>
<td>-LogFilePath &lt;path&gt;</td>
<td>Indicates the full path of the log file. The log file contains text describing all writes performed against the XenDesktop 7.x Site. Default = .\Import-XdSettings.log</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>-2 or -help</td>
<td>Displays information about the parameters and exits without importing any data.</td>
</tr>
</tbody>
</table>

If the XML file contains policy data, either all polices are imported successfully or, if there is any failure, no policy data is imported. Importing large numbers of policies with many settings can take several hours.

4. When the script completes, the message Done appears.

After successfully importing the data from the XML file, you can either run further export and import iterations, or, if you have imported all the relevant data, complete the post-migration tasks described in **Post-migration tasks**.
Migration examples

Example 1: Single large scale XenDesktop 4 farm to a XenDesktop 7 Site

In this example, the entire XenDesktop 4 farm is upgraded to XenDesktop 7.

Starting conditions

A XenDesktop 4 farm is in use. The XenDesktop 4 farm has 50 desktop groups, where each group contains an average 100 desktops in it. The XenDesktop 4 desktops are provided through Provisioning Services (PVS), and the machines are running on VMware ESX hypervisors. The VDA software installed on all the VMs is the XenDesktop 4 VDA software.

Migration steps

1. Upgrade all XenDesktop 4 VDAs to XenDesktop 5.6 Feature Pack 1 VDA software. Doing so allows the VDAs to register with both the XenDesktop 4 controller and the XenDesktop 7 Delivery Controller.
   - For Windows 7 VDAs running on XenDesktop 4, see the section "Upgrading the Virtual Desktop Agent on a VM or Blade Computer" in the XenDesktop 5.6 Feature Pack 1 product documentation located in the Citrix eDocs Archive.
   - For Windows XP and Windows Vista VDAs running on XenDesktop 4, see Install an earlier VDA on Windows XP or Windows Vista.
2. Make sure that all users log off the existing XenDesktop 4 farm.
3. Make sure that all these machines are in maintenance mode.
4. Run the export tool (XdExport) on the existing XenDesktop 4 farm.
5. Install XenDesktop 7.
   1. Perform the XenDesktop 7 Site configuration using Studio. Select the Deployment Wizard Full Production Site mode.
   2. Upgrade the Provisioning Services (PVS) server and PVS agents if PVS is part of the deployment.
   3. Upgrade the License Server (and associated licenses).
6. Unzip the Import Tool (XdImport) to a local directory on the XenDesktop 7 Delivery Controller.
7. Copy the XML file (XdSettings.xml) generated in Step 3 by the Export Tool to the local directory.
8. From the PowerShell pane of the Studio root node on the XenDesktop 7 Site, start a PowerShell session.
9. Run the Import Tool (XdImport) passing the credentials of the associated hypervisors, and the path of the XML file.
10. Manually recreate Administrator settings through Studio's Administrator node, as described in Administrators.
11. Modify the imported desktops to use registry-based Controller discovery; and point them to the new XenDesktop 7 Delivery Controller.
12. For VDAs running on Windows 7, Citrix recommends that you upgrade those VDAs to use the XenDesktop 7 Desktop OS VDA. Doing so provides access to all XenDesktop 7 features. After upgrading VDA software to XenDesktop 7 for machines in a catalog or Delivery Group, upgrade the catalog and Delivery Groups as described in Upgrade a machine catalog and Upgrade a Delivery Group.
13. Remove the Delivery Groups from maintenance mode.
14. Configure StoreFront to provide the desktops formerly provided through Web Interface. For information about installing and configuring StoreFront, see the StoreFront product documentation.

Example 2: Complete a XenDesktop 4 farm export with a partial import to XenDesktop 7.1 Site

This example shows a more commonly used method in which migration occurs in a number of steps, at each step migrating a subset of the remaining desktops needing migration.
Starting conditions

A XenDesktop 4 farm is in use, and an XenDesktop 7.1 Site has already been created and is in use. The XenDesktop 4 farm has 50 desktop groups, and each group contains an average 100 desktops. The XenDesktop 4 desktops are provided through PVS, and the machines are running on Citrix XenServer hypervisors. The version of VDA software installed on all XenDesktop 4 virtual machines is XenDesktop 4.

Migration steps

1. Run the export tool on the existing XenDesktop 4 farm.
   1. Unzip the Export Tool (XdExport) on one of the Desktop Delivery controller machines in the farm.
   2. As a Citrix Administrator, run the Export Tool with no parameters.
2. Copy and edit the resulting XML file so that it contains only the groups and desktops that you want to migrate.
3. Make sure that all users on desktops to be migrated have logged off the existing farm.
4. Make sure that all desktops that are to be migrated are in maintenance mode on XenDesktop 4.
5. Unzip the Import Tool (XdImport) to a local directory on the XenDesktop 7.1 Delivery Controller.
6. Copy the edited XML to the local directory.
7. From the PowerShell pane of the Studio root node on the XenDesktop 7.1 Site, start a PowerShell session.
8. Run the Import Tool (XdImport) passing the credentials of the associated hypervisors, and the path of the XML file.
9. Manually recreate Administrator settings through Studio’s Administrator node, as described in Administrators.
10. Modify the imported desktops to use registry-based Controller discovery; and point them to the new XenDesktop 7.1 Delivery Controller.
11. Upgrade all VDAs to the appropriate VDA software:
   - For Windows 7 VDAs:
     - Upgrade to XenDesktop 7 Virtual Delivery Agents as described in the section “Upgrading the Virtual Desktop Agent on a VM or Blade Computer” in the XenDesktop 5.6 Feature Pack 1 product documentation located in the Citrix eDocs Archive
     - After upgrading all VDA software to XenDesktop 7 for machines in a catalog or Delivery Group, upgrade the catalog and Delivery Groups as described in Upgrade a machine catalog and Upgrade a Delivery Group.
   - For Windows XP and Windows Vista VDAs, upgrade to XenDesktop 5.6 FP1 as described in Install an earlier VDA on Windows XP or Windows Vista.
12. Remove the Delivery Groups from maintenance mode.
13. Configure StoreFront to provide the desktops formerly provided through Web Interface. For information about installing and configuring StoreFront, see the StoreFront product documentation.
Post-migration tasks

Mar 28, 2015
After successfully importing data from a XenDesktop 4 farm to a XenDesktop 7.x Site, perform the following tasks before using the new Site for production work:

- Upgrade the machines' Virtual Delivery Agents (VDAs). Although it is not required, Citrix recommends that you upgrade VDAs before upgrading Controllers, Studio, or Director.
  - For Windows Vista and Windows XP, upgrade to XenDesktop 5.6 Feature Pack 1 Virtual Desktop Agent.
  - For Windows 7, upgrade to the XenDesktop 7.x Virtual Delivery Agent.
- Create any administrators you need for the XenDesktop 7.x Site.
- Update user devices — Citrix recommends that you update user devices with the latest version of Citrix Receiver to benefit from hotfixes and to receive support for the latest features. For further details, see Receiver and Plug-ins.
- Modify the imported desktops to use registry-based Controller discovery, and point them to the XenDesktop 7.x Controllers using one of the following methods:
  - Manually edit the registry:
    - Remove unnecessary Organizational Unit (OU) GUID registry entry.
    - Add a ListOfDDCs registry entry.
  - Set up a machine policy to distribute the list of Controllers to the desktops, using the Active Directory policy GPMC.msc. You cannot configure this setting from Studio.
Registry-based Controller discovery is the default for XenDesktop 7.x, but Active Directory-based discovery is still available; for further details, see Active Directory.
- Optionally, implement the following registry key settings described in the best practices for XenDesktop registry-based registration in CTX133384:
  - HeartbeatPeriodMS
  - PrepareSessionConnectionTimeoutSec
  - MaxWorkers
  - DisableActiveSessionReconnect
  - ControllersGroupGuid
If you do not perform this action, the default XenDesktop 7.x settings for these keys are used.
- Remove the imported machines from maintenance mode if they were in maintenance mode in XenDesktop 4 before the XML file was generated.
- Check the XenDesktop 7.x settings to make sure that they are correct, particularly if you had changed the PortICAConfig XML file on XenDesktop 4.
- Review all migrated components described in Data import and export details to make sure that the migration was successful.
XenDesktop 4 components that are not migrated

Mar 28, 2015
Not all XenDesktop 4 components are supported in this release. This table summarizes components that are not migrated.

<table>
<thead>
<tr>
<th>Data type</th>
<th>Notes</th>
</tr>
</thead>
</table>
| Virtual Delivery agent                 | Before a XenDesktop 7.x Delivery Controller can manage virtual desktops from XenDesktop 4, you must upgrade the VDAs to a minimum release of XenDesktop 5.x.  
For information about upgrading VDAs, see Post-migration tasks. |
| Controllers                            | You must deploy new Controller servers. You cannot upgrade a XenDesktop 4 Controller to a XenDesktop 7.x Site.  
XenDesktop 7.x Sites cannot join a XenDesktop 4 farm, and XenDesktop 4 Controllers cannot join a XenDesktop 7.x Site.  
In addition, each version has different server requirements. XenDesktop 4 requires Microsoft Windows Server 2003, and XenDesktop 7.x requires Microsoft Windows Server 2008 R2 or Microsoft Windows Server 2012.  
XenDesktop 7.1 and 7.5 also support Microsoft Windows Server 2012 R2. |
| Web Interface                          | Citrix recommends using StoreFront with XenDesktop 7.x. For information about installing and configuring StoreFront, see the StoreFront product documentation.  
When the installation program detects Web Interface, it installs StoreFront, but does not remove Web Interface. |
| Active Directory Organizational Unit (OU) configuration | Sharing an Organizational Unit (OU) between two farms or two Sites, or a farm and a Site is not supported.  
If you plan to configure the new Site to use Active Directory-based Controller discovery rather than the default registry-based Controller discovery, you must create a new OU to support it. |
<p>| PortICAConfig XML file                 | If you have changed the default settings for this file you may need to configure these settings for the new Site through Group Policy Objects. |
| Configuration logging settings provided through XenDesktop 4 Service Pack 1 |  |
| Provisioning services-related          |  |</p>
<table>
<thead>
<tr>
<th>Data type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applications</td>
<td></td>
</tr>
<tr>
<td>List of Controllers</td>
<td></td>
</tr>
<tr>
<td>NetScaler GateWay</td>
<td></td>
</tr>
<tr>
<td>Event log throttling settings</td>
<td></td>
</tr>
</tbody>
</table>
Deliver

Mar 12, 2015

This release simplifies desktop and application delivery and access while optimizing Windows applications for native look and feel. This section provides information to help you:

- Deliver desktops and applications
- Create delivery groups
- Add applications
- Use App-V
- Use Local App Access
- Deliver a high definition user experience with HDX
- Enhance user experience for mobile devices
- Enable Remote PC Access
- Prevent user access to local desktops
- Server VDI
Citrix Studio provides a variety of desktop and application delivery approaches. These options help you meet the computing requirements of every type of user in the organization.

Collections of identical virtual machines (VMs) or physical computers are managed as a single entity called a machine catalog. If you are not provisioning through Machine Creation Services or Provisioning Services, Citrix recommends that you use Microsoft System Center Configuration Manager or another third-party application to ensure that the machines in the catalog are consistent.

The machines within these machine catalogs are configured to run either a Windows Desktop OS or a Windows Server OS. For users who remotely access their office machine, you can select Remote PC Access for those machine catalogs.

Note: Amazon Web Services (AWS) only supports Server OS and Server VDI machine catalogs.
You create Delivery Groups to deliver desktops, applications, or a combination of desktops and applications to one or more users. Creating a Delivery Group is a flexible way of allocating machines and applications from a machine catalog to users.

The following figure shows Studio’s machine catalogs and Delivery Groups.
In a Delivery Group, you can:

- Use machines from multiple catalogs
- Allocate a user to multiple machines
- Allocate multiple users to one machine

As part of the creation process, you specify the following Delivery Group properties:
Users, groups, and applications allocated to Delivery Groups

Desktop settings to match users' needs

Desktop power management options

The following figure shows how users access desktops and applications through machine catalogs and Delivery Groups.

Delivery method selection

There are several ways to deliver desktops and applications. The method you choose may depend on the types of applications you are hosting, the system resources you have available, and the type of user and user experience you want to provide.

For detailed information about selecting a delivery method, see Choose an application and desktop delivery method.

The following information can help you choose between these delivery methods:

- Server OS provides access for multiple users connecting to one machine or one application.
- Desktop OS provides access for a single user connecting to a machine or application.
- Remote PC Access provides access to users' office computers from other locations through a single user connection. Remote PC Access does not require a VPN to provide security.
Set up machine catalogs

May 08, 2013

To deliver desktops and applications to users, the machine administrator creates a catalog of machines. The assignment administrator allocates machines from the machine catalog to users by creating Delivery Groups.

Cloud-provisioned machine catalogs

This release supports hosting a Site in a cloud environment managed by either Citrix CloudPlatform (CloudPlatform) or Amazon Web Services (AWS). Keep the following in mind when creating cloud provisioned machine catalogs:

- Machines are provisioned by Machine Creation Services (MCS).
- AWS only supports Server OS and Server VDI machine catalogs.
- AWS does not support Remote PC Access catalogs.

Prepare to create machine catalogs

Before you begin, ensure that you:

- Identify the types of machine catalogs to create based on your environment and users. See Desktop and application delivery.
- Identify the machine catalog machine infrastructure for your desktop delivery:
  - Power-managed
    - This machine catalog contains machines that are power-managed through Studio. If no hypervisor connections are configured when you create the machine catalog, this option is not available.
    - Connection to a cloud environment are always through virtual machines provisioned by MCS or existing images that use another service or technology option.
  - Not power-managed — This machine catalog contains machines (usually physical) that not power-managed through Studio.
- Consider how you manage and create machines; this affects the recommended machine catalog type. You manage a machine catalog's machine images through one of the following methods:
  - Machine Creation Services use a master image or template within your environment to manage virtual machines, enabling you to easily manage and update target devices through one master image or template.
  - Provisioning Services manage target devices as a device collection. A Provisioning Services vDisk imaged from a master target device delivers the desktops and applications and enables you to leverage the processing power of existing hardware or virtual machines.
  - Existing images use another service or technology option, and manage and deliver desktops and applications that you have already migrated to VMs in the data center. You must manage target devices on an individual basis or collectively using third-party electronic software distribution (ESD) tools.
- Prepare your environment:
  - Prepare a master image or template.
  - Provide Active Directory computer accounts.
  - Network interface card configuration — If multiple network interface cards (NIC) are available on a master image, you can select one to be used with the machine catalog, and you can also assign an available network to the NIC. You can also add NICs as described in Configure a network interface card.

Create machine catalogs
Use the Create Machine Catalog wizard to create new machine catalogs. The wizard prompts you about your environment and helps you select the machine catalog type that fits your requirements based on your responses.

**Test the machine catalog**

After creating or changing a machine catalog, you can run tests to ensure that the machine catalogs are ready for use.
Choose the desktop environment

May 08, 2013

The machine catalog type defines the hosting infrastructure for desktops and applications, and the level of control that users have over their environment. When deciding which type to use, consider the tasks that users perform and the devices to which the desktops are delivered.

Use the Create Machine Catalog wizard to set up machine catalogs. The wizard prompts you about your environment as described in the following sections, and based on your responses, helps you select the machine catalog type that fits your requirements.

Workspace and hardware selection

Select a machine catalog that is provides desktops based on one of the following environments.

Server OS

Server OS machine catalogs provide a Windows Server environment and provides standard desktops and applications that are shared by a large number of users.

Note: Amazon Web Services (AWS) only supports Server OS and Server VDI machine catalogs.

These machine catalogs provide desktops and applications that are:

- Allocated to users on a per-session, first-come first-served basis.
- Deployed on standardized machines.

Your users:

- Are task workers who require standardized virtual desktops and applications, such as call center operators and retail workers.
- Do not need to or are not permitted to install applications on their desktops.

You want to:

- Optimize hardware use by providing only the number of desktops that are required at any one time rather than assigning a specific desktop to each user.
- Maintain control over desktops and increase security by preventing users from making permanent changes.
- Minimize desktop management costs by providing a locked-down, standardized environment for your users.

Desktop OS

Desktop OS machine catalogs provide a Windows desktop environment and provide desktops and applications that are assigned to individual users.

Your users:

- Are task or knowledge workers who require personalized desktops of which they can take ownership.
- Are mobile workers who want to access the same desktop from a variety of devices over different networks.
- Need to install their own applications on their desktops.

You want to:
● Standardize certain aspects of users' desktops through the use of a common template.
● Deliver users' desktops to any device regardless of hardware capability.
● Reduce desktop management costs while still providing your users with a personalized desktop experience.

Remote PC Access

Remote PC Access machine catalogs provide users with remote access to their physical office desktops. You want to allow users to work at any time, anywhere, from any device.

Note: A notification displays if there are machines running the Microsoft XP operating system, or a Virtual Delivery Agent (VDA) that is earlier than XenDesktop 7. Although these machines can be included in separate machine catalogs and Delivery Groups, they cannot use some new features introduced in this release.

Machine management

Select the machine catalog machine infrastructure. The following terminology reflects choices available in the Machine Catalogs wizard, not the typical definitions. In this context, physical does not refer to a hardware-based device, and virtual is not necessarily a software implementation of a computing environment.

Using Virtual Machines

Choose this type for machine catalogs that are based on machines power-managed through Studio or provisioned through a cloud environment.

Note: If no hypervisor connections are configured when you create the machine catalog, this option is not available.

Using Physical Hardware

Choose this type for machine catalogs that are based on machines that are not power-managed through Studio.

Desktop image management

How you manage and create machines affects the recommended machine catalog type. You manage a machine catalog's machine images through one of the following methods:

Machine Creation Services use a master virtual machine within your environment to manage virtual machines, enabling you to easily manage and update target devices through one master image.

Provisioning Services manage target devices as a device collection. The desktops and applications are delivered from a Provisioning Services vDisk imaged from a master target device that enables you to leverage the processing power of existing hardware or virtual machines.

Note: Provisioning Services are not supported with CloudPlatform and AWS.

Existing images (Another service or technology option) manage and deliver desktops and applications that you have already migrated to VMs in the data center. You must manage target devices on an individual basis or collectively using third-party electronic software distribution (ESD) tools.

Desktop experience

Select the type of desktop assigned to users when they log on.

Select one of the following options:

● How many users can connect to the same machine? — Select one at a time or more than one at a time.
● I want users to connect to a new (random) desktop each time they log on — No user changes are retained.
● I want users to connect to the same (static) desktop each time they log on — Choose from the following options for
retaining changes:
- Store changes on a separate Personal vDisk
- Retain changes, and create a dedicated virtual machine and save changes on a local disk. This selection prompts you for the following user data storage information:
  - Personal vDisk size
  - Drive letter
  - Discard changes and clear virtual desktops at logoff

Virtual Machines

Number and computing power

Select the number of virtual machines to create for the catalog.

Select specifications for the virtual machines. For virtual machines that reside in an on-premise data center, select RAM and CPU settings.

For cloud environments, the computing power of a virtual machine (such as RAM and CPU settings) are called:

- Service Offering — CloudPlatform
- Instance Type — AWS

Service Offerings and Instance Types are designated as a single value, often designated as **Small**, **Medium** and **Large**. The cloud provider, not Studio, sets up how these values map to virtual machine hardware profiles.

Machine templates

For AWS environments, select the template for the virtual machines that enables you to easily manage and update target devices.
Prepare a master image

May 07, 2013

To deliver desktops and applications for machines in Server OS or Desktop OS machine catalogs, you must prepare the master image that creates the user desktops and applications.

Note: Remote PC Access machine catalogs do not require master images. The master image is a template that you use to deploy your environment. In addition to desktops and applications, creating a master image includes installing and configuring the operating system and any software you want to include on your image.

Important: If you are using Provisioning Services or Machine Creation Services, do not run Sysprep on the master images. Microsoft KMS activation with Machine Creation Services

If you are using a XenDesktop 7.x VDA with either XenServer 6.1 or XenServer 6.2, or vSphere, or Microsoft System Center Virtual Machine Manager (VMM), you do not need to manually re-arm Microsoft Windows or Microsoft Office when using Machine Creation Services (MCS).

If you are a XenDesktop 5.x VDA with XenServer 6.0.2, follow the procedure documented in CTX128580 when using Machine Creation Services (MCS).

Windows Desktop Experience with Server OS machines

You must add the Windows Desktop Experience Feature Pack to a master image for Server OS machines running on:

- Windows Server 2008 R2
- Windows 7 N edition
- Windows XP

Without this Feature Pack, Windows Media Player does not properly play video in a session. Follow the procedure documented in CTX133429.

GPU master images

For GPU-capable machines and machine catalogs, you must use a GPU-dedicated master image. GPU virtual machines require video card drivers that support GPUs and must be configured in a way that allows the VM to operate with software that uses GPU for operations. You create the master image using the XenServer's XenCenter console. See the XenServer documentation for specific details.

The following procedure describes the high-level steps to create a GPU master image that Studio detects when you select a GPU capable resources as described in Connections and resources:

1. Using XenCenter, create a VM with standard VGA, networks, and vCPU.
2. Update the VM configuration to indicate that it can use GPU (either Passthrough or vGPU).
3. Install a supported operating system in the VM.
4. Install XenServer Tools in the VM.
5. Enable RDP within the VM’s operating system.
6. Install NVIDIA drivers.
7. Turn off the Virtual Network Computing (VNC) Admin Console (through XenCenter to optimize performance) and then reboot the VM.
You are prompted to use RDP.

8. Using RDP, install XenDesktop VDA software and then reboot.
9. Optionally, create a snapshot for the VM to form a baseline that you can use as a template for other GPU master images.
10. Using RDP, install any customer-specific applications that use GPU capabilities and that are configured within XenCenter.

To prepare a master image

1. To create Server OS or Desktop OS machine catalogs, use the management tool for your hypervisor to create a new master image and install the operating system (including all service packs and updates). If they are sufficient to allow the master image to run, the number of vCPUs and the amount of memory you assign to the master image are not critical at this stage because you can change these settings when you create the machine catalog.

   Important: Make sure that you set up the master image with the same amount of hard disk space that is required for users' desktops and applications because this value cannot be changed later. Make sure that the hard disk for the master image is attached at device location 0. Most standard master image templates configure this location by default, but some custom templates may not do so.

   For machines provisioned through Provisioning Services (PVS), you can either use a master image or a physical computer as your master target device.

2. On the master image, install the appropriate integration tools for your hypervisor (XenServer Tools, Hyper-V Integration Services, or VMware tools).

   Note: If you do not install hypervisor integration tools on the master image, your desktops may not function correctly.

3. Install and reconfigure the Virtual Delivery Agent (VDA) from the installation media as described in Install using the graphical interface. When installing the VDA, select the option to optimize the desktop. This improves the performance of users' desktops and applications by reconfiguring various Windows features that are incompatible with or unnecessary for virtual desktops.

4. Install any third-party tools that you want to run, such as anti-virus software or electronic software distribution agents, and configure services such as Windows Update, as required for your deployment. Make sure that you use settings appropriate for your users and the machine type you intend to use, as these configurations are propagated to users' desktops and applications from the master image.

5. Install and configure any third-party applications that you do not want to virtualize. Citrix recommends virtualizing applications. This approach significantly reduces desktop management costs by removing the need to update the master image whenever you want to add or reconfigure an application. In addition, with fewer applications installed on each desktop, you can reduce the size of the master image hard disks to save on storage costs.

6. If you are planning to publish App-V applications through Studio as described in Manage App-V application delivery, install and configure App-V clients with all the recommended App-V settings on the master image.

7. For machine catalogs created through Machine Creation Services (MCS), if you need to localize Microsoft Windows, install the locales and language packs on the master image before provisioning. During provisioning, when a master image snapshot is created, the provisioned VMs use the installed locales and language packs.

8. To deliver Server OS or Desktop OS machine desktops, join the master image to the domain for which you want users' desktops and applications to be members, and make sure that the master image is available on the host where you want to create the machines.

   Citrix recommends that you create a snapshot of your master image and name the snapshot so that you can identify the master image in the future. If you specify a master image rather than a snapshot when creating a Server OS or
Desktop OS machine catalog, Studio creates a snapshot for you but you cannot name it.

For machine catalogs created through Provisioning Services as described in Create a new machine catalog, create a VHD file for the vDisk from your master target device before you join the master target device to a domain. For more information about imaging a vDisk, see Installing and Configuring Provisioning Services.
Provide Active Directory computer accounts

May 10, 2013

Each machine in a machine catalog needs a corresponding Active Directory computer account. If you plan to create random or static machine catalogs and you have access to an Active Directory domain administrator account, you can allow Studio to create new accounts when you create the machine catalog. If you do not have the necessary permissions, ensure that you have a sufficient number of unused Active Directory computer accounts available for the machines before you start the Create Machine Catalog task.

You can select the existing computer accounts to use by browsing the Active Directory when you create the machine catalog or, alternatively, you can import a .csv file containing a list of account names. Studio requires the following format for computer accounts imported from .csv files.

[ADComputerAccount]
ADcomputeraccountname.domain
...

For existing and physical machine catalogs, you select or import existing accounts and assign each virtual or physical machine to both an Active Directory computer account and to a user account. For machines created through Provisioning Services, Active Directory computer accounts for target devices are managed using Provisioning Services and existing Active Directory tools. For more information about Active Directory integration with Provisioning Services, see Provisioning Services.

For information about managing a machine catalog's Active Directory or Organizational Units (OU) accounts, see Manage desktop computer accounts.
Create a new machine catalog

Aug 29, 2013

The Create Machine Catalog wizard guides you through the machine catalog creation process. Machine catalog creation is the first step in delivering desktops to users.

Steps for creating machine catalogs vary based on choices you make for the tasks that users perform, the devices to which the desktops and applications are delivered, or user experience requirements.

The major differences result from your choice of one of the following machine catalog types:

- Server OS is appropriate for users who perform a set of well-defined tasks, and who do not require personalization for their work.
- Desktop OS lets you provide individual desktop environments for each user as well as customizable desktops, including Personal vDisks (PVD).
- Remote PC Access lets your users remotely connect to their office computers through a secure connection.

Before you start the Create Machine Catalog task, make sure that you have all the prerequisites in place for the particular type you intend to use.

<table>
<thead>
<tr>
<th>Machine catalog type</th>
<th>Requirement</th>
</tr>
</thead>
</table>
| All                                          | • A host with sufficient processors, memory, and storage to accommodate the number of machines you plan to create.  
• If there are multiple Network Interface cards (NICs), know which virtual networks you want to assign to each card. |
| Desktop OS and Server OS                     | • Either a sufficient number of unused Active Directory computer accounts for the machines you plan to create, or access to an Active Directory domain administrator account for the domain on which you want the desktops to be members.  
• A master Virtual Machine image (master image) from which to create the desktops. The master image must be available on the host where the machines are created. |
| Remote PC Access                             | Specify Machine Accounts or Organizational Units. These features let you add the machines corresponding to users or user groups. |
| Existing                                     | • Virtual Machines or dedicated blade PCs hosting user desktops that you have already migrated to the data center.  
• Active Directory user and computer accounts to assign to the Virtual Machines or blade PCs. |
| Desktop OS machine catalog that allows customization | • A Provisioning Services deployment with a Personal vDisk that you have imaged from the master target device.  
• Device collections configured to load the Personal vDisk over the network.  
• Active Directory computer accounts managed by Provisioning Services for each target device |
<table>
<thead>
<tr>
<th>Machine catalog type</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>in the device collections.</td>
</tr>
<tr>
<td></td>
<td>Select the kind of user experience you want to provide:</td>
</tr>
<tr>
<td></td>
<td>● Connect to a random desktop or the same desktop on logon.</td>
</tr>
<tr>
<td></td>
<td>● Discard or retain changes. If users retain changes, plan on where to save changes (on a separate Personal vDisk or on a local disk)</td>
</tr>
<tr>
<td>Server OS</td>
<td>Active Directory user and computer accounts to assign to the Virtual Machines or blade PCs.</td>
</tr>
</tbody>
</table>

To create a machine catalog

1. Log on to the computer running Studio. If you plan to use XenDesktop to create new Active Directory computer accounts for the machines, log on using a domain administrator account for the domain on which you plan to add the desktops.

2. Select the Machine Catalogs node and click Create Machine Catalog.

   If the Machine Catalogs node is not visible, you must complete one of the initial configuration tasks presented when you first start Studio before you begin creating the first machine catalog.

3. Select the machine image management method:
   ● Machine Creation Services (MCS)
     The options to use virtual machines and Machine Creation Services are disabled if you do not have access to or have not configured a hosting infrastructure. For information about setting up the hosting infrastructure, see Connections and resources.
     To create a machine catalog that uses GPU resources, you can only use MCS to create a GPU-enabled master image.

   ● Provisioning Services — Select the Provisioning Services server by entering the IP address and domain.
   ● Another service or technology (existing images)

4. On the Desktop Experience page (which is not available for Remote PC Access), choose how users connect each time they log on:

   ● For Desktop OS machines:
     ● Select random desktop or to the same static desktop
     ● Select whether to retain changes and how to save the changes: Personal vDisk or dedicated virtual machine

   ● For Desktop OS machines with Provisioning Services — Select random desktop or to the same static desktop

5. On the Machines page, add computer accounts:
   ● Active Directory Computer Accounts for Desktop OS and Server OS machine catalogs
   ● Organizational Units (OU) for Remote PC Access machine catalogs

8. On the Users page, if you have machines in the machine catalog that are running Windows XP or Windows Vista, select the Some of the machines are running Windows XP or Windows Vista option.

9. If you selected a machine image management method that uses a master image, on the Master Image page, select a
host and a corresponding snapshot or virtual machine. To create a machine catalog with GPU capabilities, select a snapshot or master image that has a compatible GPU resource.

10. If you selected a machine image management method that requires virtual machines, on the Add and Configure Virtual Machines page:
   - Select the number of virtual machines.
   - Configure the number of virtual CPUs, memory size, and hard disk size.
   - Specify the Personal vDisk size and drive letter

11. On the Scopes page, define which administrators can access the machine catalog.

12. On the Summary page, add descriptions and create the machine catalog.

You have now created a machine catalog that provides the machines on which users’ desktops reside.

To deliver desktops from the machines in your machine catalog to users, you must allocate the machines to users by creating Delivery Groups. For more information, see Create a new Delivery Group.
Configure a network interface card

May 10, 2013
When creating machine catalogs, if the master image has multiple network interface cards, configure one or more cards to work with networks. For example, if there are two cards on the master image, you can configure one of the cards to access a specific secure network, while the other accesses a more commonly used network.

1. In the Create Catalog wizard, select the Network Cards Configuration node.
2. Select a card in the display and make sure it is enabled.
3. Select one or more virtual networks to associate with the selected card.
4. Optionally add or remove cards and then click Next to continue to create the machine catalog.
Test a machine catalog

May 10, 2013
To make sure your machine catalog is properly configured to deliver desktops or applications, you can test the machine catalog. Machine catalog configuration testing occurs:

- Automatically after you create a machine catalog
- By clicking Test Machine Catalog:
  - In the Common Tasks page
  - In the Quick Launch page
  - As an action performed on a machine catalog in the Machine Catalogs page

1. If you have not already done so, click Studio to display the console.
2. Select a machine catalog and then click Test Catalog.
   When the test completes:
   - Optionally view a test report
   - Resolve any issues and then test again
   - Continue setting up other Studio components
Delivery Groups

May 10, 2013

Delivery Groups are collections of machines, and specify who can use a group of desktops or applications. Create Delivery Groups for specific teams, departments, or types of users. With Delivery Groups, you can:

- Specify groups of users who access desktops, applications, or desktops and applications
- Add users and groups of users

**Delivery Group users**

To deliver applications and desktops, you add users or user groups to Delivery Groups. You can select user groups by browsing or entering a list of Active Directory users and groups.

For Desktop OS Delivery Groups, you can import user data from a file after you create the group.

**Delivery Group user experience**

A Delivery Group's characteristics are based on the desktops or machines within the group's machine catalog. A desktop is a virtual or physical machine that is created manually through the Create Machine Catalog wizard. For Server OS and Desktop OS machine catalogs, they can also be provisioned automatically using Machine Creation Services or Provisioning Services. The desktop characteristics come from the machine catalog to which they are assigned.

A machine catalog is a collection of physical computers and virtual machines that you assign to users through the Delivery Group. What the user sees depends on the machine type:

- Server OS desktops and applications — Users experience a Windows server environment. This type lets multiple user sessions share a single Windows Server environment. This option is appropriate for users who perform well-defined tasks, and do not require personalization. This is a random desktop environment that let users connect to any available Windows server environment.
- Desktop OS desktops and applications — Users experience a Windows client environment. This type lets multiple users log in to one Windows client environment. This option provides the following desktop environments:
  - Static environment for users and lets them connect to a personalized desktop or application
  - Random environment in which the machine state is reset on user logoff, and are available for other users
  - Remote PC Access machines that provide users with remote access to their office desktops.

Neither machine catalogs nor Delivery Groups can contain a mixture of these types; they must include either all Server OS, Desktop OS, or Remote PC Access machines.

**Delivery types**

Depending on your environment as described in "Delivery Group user experience", you can select a Delivery Groups type that determines what Delivery Groups provide to user devices:

- Desktops only
- Applications only
- Desktops and applications (not available for static Desktop OS machines)
Personalize desktops for users

May 10, 2013

The example scenario in this topic describes the major steps to follow to provision and deliver two Delivery Groups. One Delivery Group is based on a Windows Server 2008 R2 master image but has a Windows 7 look and feel. The other is based on a Windows 8 master image. The following personalization capabilities ensure a consistent logon experience and customizable applications for users in those Delivery Groups:

- Policies and Active Directory Group Policy Objects provide global management of user personalization settings across both Delivery Groups.
  For simple scenarios, you might consider using Citrix Profile management, which is installed silently on master images when you install the Virtual Delivery Agent. It lets you define profile behavior on a per-Delivery Group basis.
- The Personal vDisk feature retains the single image management of pooled desktops while allowing users to install applications and change their desktop settings. This feature is used for the Delivery Group that is based on a Windows 8 image.

Desktop strategy

A new corporate desktop strategy means you want to virtualize as many corporate desktops as possible. In this example, you begin with the desktops in two of your organization's teams.

- The Accounts team use a large number of physical workstations installed with several standard financial applications. The workstations run Windows 7. To replace these with virtual desktops, create a Windows Server 2008 R2 Delivery Group with a Windows 7 look-and-feel for this team.
  Members of the Accounts team do not need to install their own applications, so they do not require Personal vDisks. However, they typically customize the applications on their physical machines (for example, select a home page). To preserve personalized settings in your new virtualized environment use profile management policies.
- The Sales team are eager to upgrade to Windows 8 on workstations that they use to test their customers' applications. The team must be able to install and uninstall the desktops themselves, so use the Personal vDisk feature to enable that capability.

Desktop environment

The sample scenario assumes the following about your environment:

- Your network has fast, low latency connections
- You have estimated the space needed for Personal vDisks based on the actual and expected levels of personalization in the enterprise

To implement the new desktop strategy effectively, use these required components:

- A supported hypervisor
- XenApp or XenDesktop
- A provisioning solution:
  - Machine Creation Services (MCS) is available by default if, when creating an application and desktop delivery site, you configure a Host and specify that MCS is to create VMs.
  - Use of MCS provides consolidated administration through Citrix Studio.
Citrix Provisioning Services (PVS) is another option for use with System Center 2012 Virtual Machine Manager. To prepare for Provisioning Services use, you must separately install the Provisioning Services server and configure a Provisioning Services master vDisk image. VDA installation installs the Provisioning Services agent on the master images for Windows 7 and Windows 8. You must use the Provisioning Services console to manage Provisioning Services.

Note: A provisioning solution is not required to deploy unmanaged physical desktops.

- Master image for Windows Server 2008 R2 and for Windows 8
- Citrix Receiver, installed on a user device to test access to your new virtual desktops

**Studio**

Studio is the management console that enables you to configure and manage your deployment, eliminating the need for separate management consoles for managing the delivery of applications and desktops.

Studio is a Microsoft Management Console snap-in that displays all of the objects in your deployment. It provides various wizards to guide you through setting up your environment, creating workloads to host applications and desktops, and assigning applications and desktops to users.

**Configuration**

After installing required components, configure your environment using the following general steps. For detailed instructions, see Create a site.

1. To ensure business continuity, enable the AlwaysOn Availability Groups feature in SQL Server 2012.
2. After you install the mirror database software on a different server, use Studio to create a full deployment Site on the server where you installed the Delivery Controller. Depending on your database permissions, Studio will configure the principle and mirror Site Configuration Databases, or you can generate scripts that your database administrator can use to configure them. The Controller then automatically recognizes that the Site Configuration Database is replicated and uses the mirror if necessary.
   When creating the site, select the storage location for Personal vDisks. Although you store Personal vDisks (.vhd disks) physically on the hypervisor, they do not have to be in the same location as other disks attached to the virtual desktop. This can reduce the cost of Personal vDisk storage.
3. Use the Create Machine Catalog wizard to create two catalogs for your two master images:
   - A Server OS machine catalog for the Accounts team's Windows Server 2008 R2 desktops.
     If using Machine Creation Services: Because users do not need to connect to the same instance of a desktop each time they log on, specify virtual machines of the random type.
   - A Desktop OS machine catalog for the Sales team's Windows 8 desktops.
     If using Provisioning Services: Create the catalog by connecting to a Provisioning Services server and selecting a suitable device collection. Because users want a highly personalized desktop, specify virtual machines of the static type to which you attach Personal vDisks. Also specify the size of the vDisks and the drive letter to use for them.
     Tip: Remember to run the Personal vDisk inventory if you change the master image used in the Desktop OS machine catalog. When you do this depends on your Provisioning Services setup. For more information, see Provisioning Services.
4. Using the Create Delivery Group wizard, create two Delivery Groups for your two catalogs:
   - One containing desktops and applications for the Accounts team
   - One containing desktops for the Sales team
To each Delivery Group, add accounts for the users in each team (so the right people can access the right desktops). Also add a test user account for each team.

To ensure a consistent experience when an Accounts user connects to different, randomly assigned desktops in that team's catalog, use profile definition policies to specify features such as the location of any redirected folders.

Testing

Test the user experience as follows:

1. Confirm that your new desktops are registered with the Controller by selecting your Delivery Groups in Studio and clicking Test Delivery Group. Follow the advice in any errors that are displayed.

2. From the user device, log on to Receiver as a member of Accounts. Is your Windows 7 desktop displayed correctly in the Desktop Viewer? Does it look and behave like a Windows 7 desktop even though it was created from a Windows Server machine? Change an application setting or preference that will be saved in your profile; then log off and on again. Is the new setting or preference saved?

3. Repeat the last step, logging on to Receiver as a member of Sales. Is your Windows 8 desktop displayed correctly in the Desktop Viewer? Do you have a drive (the Personal vDisk) that has the letter you specified while creating the machine catalog for this desktop? Can you install applications and save data on that drive?
Create a new Delivery Group

Mar 28, 2015

Delivery Groups specify which users can access desktops or applications, usually based on user characteristics, such as the types of users. Before creating a Delivery Group, note the following:

- You can only create a Delivery Group if at least one machine remains unused in the machine catalog you select.
- You cannot use a machine in more than one Delivery Group.
- You can create Delivery Groups from multiple machine catalogs with the same characteristics.
- A machine catalog can be associated with one or more Delivery Groups.
- Multiple catalogs can reference the same Delivery Group.
- In Studio, you cannot create mixed Delivery Groups from machine catalogs with different machine types. Machine catalog characteristics must match if you want to put the machines into a single group. For example, you cannot mix machines from:
  - Server OS machine catalogs with Desktop OS machine catalogs
  - Allocation type static machine catalog with random machine catalog
- For Remote PC Access Delivery Groups, any machine added to a Remote PC Access machine catalog automatically is associated with a Delivery Group.

To create a Delivery Group

1. In Studio, select the Delivery group node and click Create Delivery Group.
2. Click Add Machines, select a machine catalog for this Delivery Group, and then enter the number of machines the group consumes from the machine catalog.
   Tip: The total number of available machines in the machine catalog appears in the display.
3. On the Users page, click Add users to add the users or user groups that can access the desktops or applications. You can select user groups by browsing or entering a list of Active Directory users and groups each separated by a semicolon. For Desktop OS Delivery Groups, you can import user data from a file after you create the group.
4. On the Delivery Type page, select what the desktops deliver to users:
   - Desktops only
   - Applications only
   - Desktops and applications (Not available for static Desktop OS machines)
5. If the Delivery Group provides applications, on the Applications page, select applications from the display, or add applications as described in Create a Delivery Group application.
6. On the StoreFront page, select StoreFront URLs to be pushed to Citrix Receiver so that Receiver can connect to a StoreFront without user intervention. Note that this setting is for Receiver running on VDAs.
   - Select Automatic to select server URLs from the list.
   - Select Manually to enter a server address if it does not appear in the list.
   - Select No to omit adding StoreFronts and go to the Summary page.
   - Select Yes to select from existing store URLs listed in the Selected stores display.
     You can also add a new store by clicking Add new and entering the StoreFront’s URL if you do not see the StoreFronts you want in the display, and you know the store exists.
7. On the Scopes page, define which administrators can access the Delivery Group.
8. On the Summary page, check all details and then enter a display name that users and administrators see and a descriptive Delivery Group name that only administrators see.
You have now created a delivery group. For information about applications that Delivery Groups can deliver to end user devices, see Hosted applications.
Test a Delivery Group

Mar 28, 2015

Delivery Group configuration testing occurs:

- Automatically after you create a Delivery Group
- By clicking Test Delivery Group:
  - In the Common Tasks page
  - As an action performed on a Delivery Group in the Delivery Groups page

1. If you have not already done so, click Studio to display the console.
2. Select a delivery group and then click Test Delivery Group.
   When the test completes:

   - Optionally view a test report.
   - Resolve any issues and then test again.
   - Continue setting up other Studio components.
Hosted applications

Mar 28, 2015

Studio lets you provide applications to users' devices through various delivery methods.

Delivery methods include:

- Hosted applications in your datacenter as described in Create a Delivery Group application. These applications are:
  - Installed and run on physical or virtual machines.
  - Provided through a master image that creates user desktops. The master image contains those elements that are common to all users, such as anti-virus software, Citrix plug-ins, and other default programs.
- Local Access App (LAA) — Applications that are accessed by publishing shortcuts to locally-installed applications on a virtual desktop as described in Local App Access.
- Microsoft Application Virtualization applications (App-V) — Deploys App-V applications from a virtual application server as hosted applications to user devices. Once configured as described in Configure App-V resources, these applications are available to add to application Delivery Groups or desktop and application Delivery Groups. Note the following:
  - All versions prior to App-V 5.0 are not supported.
  - The App-V 5.0 client does not provide offline access to applications.
  - Use the following version combinations for App-V and XenDesktop or XenApp components:

<table>
<thead>
<tr>
<th>App-V version</th>
<th>XenDesktop or XenApp versions</th>
<th>Virtual Delivery Agent (VDA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0 R5TM</td>
<td>XenDesktop 7.0, 7.1, 7.5</td>
<td>7.0, 7.1, 7.5</td>
</tr>
<tr>
<td></td>
<td>XenApp 7.5</td>
<td></td>
</tr>
<tr>
<td>5.0 Service Pack 1</td>
<td>XenDesktop 7.0, 7.1, 7.5</td>
<td>7.0, 7.1, 7.5</td>
</tr>
<tr>
<td></td>
<td>XenApp 7.5</td>
<td></td>
</tr>
<tr>
<td>5.0 Service Pack 2</td>
<td>XenDesktop 7.0, 7.1, 7.5</td>
<td>7.1, 7.5</td>
</tr>
<tr>
<td></td>
<td>XenApp 7.5</td>
<td></td>
</tr>
</tbody>
</table>

The following diagram shows application delivery in Studio.
Once selected and installed, the applications are delivered through hosted application and desktop Delivery Groups. When a user requests an application, the application is installed on and runs on a machine associated with an application Delivery Group or desktop and application Delivery Group to which the user is assigned, as described in Create a Delivery Group application. Machines within a Delivery Group (and their associated applications) are either randomly assigned to users when they log on or statically assigned to the same user each time that user logs on. The machines hosting applications are configured as:

- **Server OS** — Capable of hosting both desktops and applications from the same machine at the same time for multiple, simultaneously connected users.
- **Desktop OS** — Capable of delivering a desktop or a set of applications from a single machine to a single user at any one time.
Because applications run on machines in the datacenter, processing on user devices is kept to a minimum. Hosting applications on desktops in your datacenter simplifies application management and provides a cost-effective application delivery solution. Applications and desktops remain in the datacenter, providing added security and a consistent user experience.

- Remote PC Access Delivery Group— Users log on remotely to their physical PC on which applications reside. The Citrix Receiver running on the client device provides access to all of the applications and data on the remote PC.

Note: To publish applications or to host Microsoft Remote Desktop Service (RDS) desktops, you must install a Virtual Delivery Agent (VDA) for Server OS on the hosting server.

Users interact with applications as if the applications were installed locally. To access applications, users click the application icon or shortcut on their devices, and the application displays in high definition. When users log off, their changes are discarded unless they have a Personal vDisk, a statically assigned desktop, or both.

In addition to the application delivery method, the application's property settings also affect the user experience. These settings are configured when you add an application to a new or existing Delivery Group.
Configure App-V resources

Mar 28, 2015

With Microsoft Application Virtualization (App-V), you can deploy applications from a virtual application server to any user device. To provide App-V applications to users:

- Configure App-V resources in Studio.
- Add App-V applications to Delivery Groups as described in Create a Delivery Group application.
- Manage the applications as described in Modify and manage applications.

1. In Studio, in the Configuration node, select App-V Publishing.
2. Select Add App-V Publishing.
3. In the App-V Settings dialog box, enter the URLs for the following servers:
   - App-V management server
   - App-V publishing server
4. Select Test connection to make sure you can access the servers, and then click Save.

You can add or change App-V servers, refresh the application display, or delete applications.

1. In the Configuration node, select App-V Publishing.
2. You can perform the following actions:
   - Edit App-V publishing settings — Add or change App-V management and App-V publishing servers.
   - Remove App-V publishing settings — Delete App-V management and App-V publishing servers.
   - Refresh App-V Applications — Refresh the App-V applications display. If applications are no longer available, or if there is a problem connecting to a server, this is indicated in the display.

Now that App-V applications are available, assign the applications to users as described in Create a Delivery Group application.
Microsoft Application Virtualization (App-V) integration

Jun 11, 2013

Microsoft Application Virtualization (App-V) lets you deploy, update, and support applications as services. Users access applications without installing them on their own devices. Applications and user settings are preserved whether users are online or offline.

This release supports App-V 5.0.

The App-V 5.0 client does not support offline access to applications.

The App-V 4.6 2 client is no longer supported.

App-V and Microsoft User State Virtualization (USV) provide access to applications and data, regardless of location and connection to the Internet.

App-V integration support includes using SMB shares for applications; the HTTP/HTTPS protocol is not supported.

Microsoft App-V components

See App-V 5.0 Supported Configurations for system requirements.

With App-V, individual applications are transformed from locally installed products into centrally managed services. Applications are available seamlessly without any pre-configuration or changes to operating system settings. App-V contains the following components:

- **Management Server** — Provides a centralized console to manage App-V 5.0 infrastructure and deliver virtual applications to both the App-V Desktop Client as well as a Remote Desktop Services Client. The App-V Management Server authenticates, requests, and provides the security, metering, monitoring, and data gathering required by the administrator. The server uses Active Directory and supporting tools to manage users and applications.

- **Publishing Server** — Provides App-V Clients with applications for specific users, and hosts the virtual application package for streaming. It fetches the packages from the management server.

- **Client** — Retrieves virtual applications, publishes the applications on the client, and automatically sets up and manages virtual environments at runtime on Windows devices. The App-V Client is installed on the VDA and stores user-specific virtual application settings, such as registry and file changes in each user's profile.

Studio components

You provide App-V applications to users in Studio using the following components:

- **Studio App-V publishing** — Configures access to App-V Publishing and Management servers so that Studio can discover and use App-V applications.

- **Machine catalogs** — Collections of physical computers and virtual machines on which applications reside. The machines and their resources are assigned to users through Delivery Groups. During machine catalog creation, the App-V Client must be installed on the master image, and configured through settings such as ShareContentStoreMode and EnablePackageScripts.

  Note: You do not need to configure the App-V Publishing Server in the master image, since it is configured during application launch.
• **Delivery Groups** — Provides access to applications for specific groups of users, such as departments or teams.

• **Applications** — Delivered to users by assigning them to Delivery Groups
Manage App-V application delivery

Mar 28, 2015

With App-V, you can deploy applications from a virtual application server to any user device.

This release supports App-V 5.

To deliver these types of applications, you must:


2. Publish the App-V applications on the App-V Management Server. Configure settings such as permissions and File Type Association. These settings are pre-existing if you already deployed App-V.

3. Install the App-V client on VDAs.

4. Install the App-V client in the master image for Studio machine catalogs.
   Note: During machine catalog creation, the App-V Client must be installed on the master image, and configured through settings such as ShareContentStoreMode and EnablePackageScripts. You do not need to configure the App-V Publishing Server in the master image since it is configured during application launch.

5. During Site configuration in Studio, add the App-V Publishing and Management server addresses. These servers are automatically obtained by the Delivery Groups in that site.

6. Use Studio to connect to App-V servers as described in Configure App-V resources.

7. Use Studio to provide App-V applications through application Deliver Groups, as described in Create a Delivery Group application.

8. The applications are now available for launch through Citrix Receiver, and are accessible through StoreFront.

The following figure illustrates the processes for delivering App-V applications.
To change publishing server settings on a VDA, Citrix recommends using the SDK cmdlets on the controller.

To view publishing server settings, enter:

```
Get-CtxAppvServerSetting -AppVPublishingServer <pubServer>
```

The following cmdlet changes the settings of the publishing server on the Delivery Controller. Not all parameters are mandatory.

```
Set-CtxAppvServerSetting –AppVPublishingServer <pubServer> -UserRefreshOnLogon <bool> -UserRefreshEnabled <bool> -UserRefreshInterval <int> -UserRefreshIntervalUnit <Day/Hour> -GlobalRefreshOnLogon <bool> -GlobalRefreshEnabled <bool> -GlobalRefreshInterval <int> -GlobalRefreshIntervalUnit <Day/Hour>
```

To make sure that App-V 5 applications launch properly, in PowerShell enter the following SDK cmdlet:

```
Set-CtxAppvServerSetting –UserRefreshOnLogon 0
```

Note: If you previously used GPO policy settings for managing publishing server settings, the GPO settings override any App-V integration settings, including the previous cmdlet settings. This may result in App-V application launch failure. Citrix recommends that you remove all GPO policy settings and configure the same settings using the SDK, as described in the SDK cmdlet documentation.

You can launch App-V applications (from Server OS and Desktop OS Delivery Groups) as described in the following use cases:

- Through Citrix Receiver
- From the Start menu
- Through the App-V client and Citrix Receiver
- Simultaneously by multiple users on multiple devices
- Through Citrix StoreFront

Note the following:

- Modified App-V application properties are implemented when the application is started. For example, for applications with a modified display name or customized icon, the modification appears users start the application.
- There is no change in App-V applications performance when a desktop and application Delivery Group is converted to an application Delivery Group.
- Only App-V server-based deployment in which an administrator uses an App-V Management Server and Publishing Server to manage App-V applications is supported.
Troubleshoot App-V

Nov 04, 2014

Test connection operation fails

If the Test connection operation returns an error when you specify App-V management server and publishing server addresses in Studio, check the following:

1. The App-V server is powered on: either send a Ping command or check the IIS Manager (each App-V server should be in a Started and Running state).
3. The App-V server is added to Active Directory.
   If the Studio machine and the App-V server are in different Active Directory domains that do not have a trust relationship, from the PowerShell console on the Studio machine, run winrm set winrm/config/client '@{TrustedHosts="<App-V server FQDN>"}': If TrustedHosts is managed by GPO, the following error message will display: “The config setting TrustedHosts cannot be changed because use is controlled by policies. The policy would need to be set to “Not Configured” in order to change the config setting”. If this message displays, add an entry for the App-V server name to the TrustedHosts policy in GPO (Administrative Templates > Windows Components > Windows Remote Management (WinRM) > WinRM Client).
4. The Studio administrator is also an App-V server administrator.
5. File sharing is enabled on the App-V server: enter \<App-V server FQDN> in Windows Explorer or with the Run command.
6. The App-V server has the same file sharing permissions as the App-V administrator: on the App-V server, add an entry for\<App-V Server FQDN> in Stored User Names and Passwords, specifying the credentials of the user who has administrator privileges on the App-V server. For guidance, see http://support.microsoft.com/kb/306541.

App-V application discovery fails

Check the following if App-V application discovery fails:

1. Studio administrator is an App-V management server administrator.
2. The App-V management server is running. Check this by opening the IIS Manager and make sure the server is in a Started and Running state.
3. PowerShell remoting is enabled on the App-V management server and on the publishing server. If either is not enabled, follow the procedure in http://technet.microsoft.com/en-us/magazine/ff700227.aspx to enable the server.
4. Packages have appropriate security permissions so that the Studio administrator has access.

Check the following if App-V applications fail to launch:

1. The publishing server is running. Check this by opening the IIS Manager and make sure the server is in a Started and Running state.
2. App-V packages have appropriate security permissions so that users have access.
3. Check the following on the VDA:
   - Make sure that Temp is pointing to the correct location, and that there is enough space available in the Temp directory.
- Make sure that App-V client is installed, and no earlier than version 5.0.
- Make sure you have Administrator permissions, run Get-AppvClientConfiguration and make sure that EnablePackageScripts is set to 1. If it is not set to 1, run Set-AppvClientConfiguration -EnablePackageScripts $true. Citrix recommends that you perform this step when you create a master image so that all VDAs created from the master image have the correct configuration.

Access the Registry editor (regedit) and go to HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Citrix\AppV. Make sure that the AppVServers key has the following value format: AppVManagementServer+metadata;PublishingServer. For example:
  - http://xmas-demo-appv.blrstrm.com+0+0+0+1+1+1+0+1;
- Make sure that CtxAppVCOMAdmin has administrator privileges. During VDA installation CtxAppVCOMAdmin is usually created and added to the Local Administrators Group on the VDA machine. However, depending on the Active Directory policy, this user might lose the administrative association. Run compmgmt.msc and browse to Local Users and Groups > Users. If CtxAppVCOMAdmin is not an administrator, fine tune the group policy or contact your administrator, so that this user account retains its administrative association.

4. On the master image on which the App-V client is installed, the administrator must set the PowerShell ExecutionPolicy to RemoteSigned. This setting is required because the App-V client module provided by Microsoft is not signed, and this setting allows PowerShell to run unsigned local scripts and cmdlets. Set the ExecutionPolicy using one of these methods:
   - Logged in as administrator, run the PowerShell command Set-ExecutionPolicy RemoteSigned.
   - From Group Policy settings, go to Computer Configuration > Policies > Administrative Templates Windows Components > Windows PowerShell > Turn on Script Execution.

5. Check the publishing servers:
   - Run Get-AppvPublishingServer * to display the list of publishing servers.
   - Check whether UserRefreshOnLogon is set to False. If it is not set to False, the first App-V application launch typically fails.
   - Note: Beginning with XenDesktop release 7.1 or XenApp 7.5, you can view and set UserRefreshOnLogon (and other publishing servers parameters) using SDK cmdlets.

If these steps do not resolve the issues, the Administrator should enable and examine the logs.

If you upgrade an App-V package containing applications published through Studio, delete the old applications, which will remove them from the application Delivery Group. Then, discover and add the new applications to the Delivery Group. For example, if you published Word 2010 and Excel 2010 from an Office 2010 App-V package and then upgraded that package to Office 2013, remove Word 2010 and Excel 2010 and then discover and add Word 2013 and Excel 2013.

Enable logs on Studio and the VDA to help troubleshoot App-V.

**To enable Studio logs**

1. Create the folder C:\CtxAppvLogs.
2. Go to C:\ProgramFiles\Citrix\StudioAppVIntegration\SnapIn\Citrix.Appv.Admin.V1, and open CtxAppvCommon.dll.config
To enable VDA logs

1. Create the folder C:\CtxAppvLogs.
2. Go to C:\ProgramFiles\Citrix\ Virtual Desktop Agent, and open CtxAppvCommon.dll.config in a text editor such as notepad as an administrator.
3. Uncomment the following line in CtxAppvCommon.dll.config:
   `<add key ="LogFileName" value="C:\CtxAppvLogs\log.txt"/>
   The logs are available at that location.

4. Uncomment the following line and set the value field to 1 as shown in the following example:
   `<add key ="EnableLauncherLogs" value="1"/>

   - All configuration related logs are available in C:\CtxAppvLogs.
   - The application launch logs are available as follows:
     - XenDesktop 7.1, XenDesktop 7.5, or XenApp 7.5 — %LOCALAPPDATA%\Citrix\CtxAppvLogs.
     - XenDesktop 7.0 — %LocalAppData%\temp\CtxAppvLogs.
     - LOCALAPPDATA resolves to the local folder for the logged in user. Make sure to check in the local folder of the launching user (for whom application launch failed):
       - Beginning with XenDesktop 7.1 or XenApp 7.5 — %LocalAppData%\Citrix\CtxAppVLogs.
       - XenDesktop 7.0 — %LocalAppData%\temp\CtxAppVLogs.

5. As Administrator, restart the Broker service or restart the VDA machine to start logging.
Local App Access

Jun 11, 2013
Local App Access seamlessly integrates users' locally installed Windows applications into a hosted desktop environment without changing from one computer to another. With Local App Access, you can:

- Access applications installed locally on a physical laptop, PC, or other device directly from their virtual desktop.

  Provide a flexible application delivery solution. If users have local applications that you cannot virtualize or that IT does not maintain, those applications still behave as though they are installed on a virtual desktop.

- Eliminate double-hop latency when applications are hosted separately from the virtual desktop by putting the shortcut to the published application on the user's Windows device.

  Use applications such as:
  - Video conferencing software such as GoToMeeting.
  - Specialty or niche applications that are not yet virtualized.
  - Applications and peripherals that would otherwise transfer huge amounts of data from a user device to a server and back to the user device, such as DVD burners and TV tuners.

Perform the following procedures to set up Local App Access.

- Activate the Local App Access feature as described in Provide access to local applications.
- Activate Local App Access on Client Machines Running Receiver as described in Enable Local App Access on client machines running Receiver.
Local App Access requirements

Jul 05, 2013

Local App Access is supported on Desktop OS machine, Server OS machine, and Remote PC Access desktops.

To enable Local App Access, the hosting environment must include the following components:

- XenDesktop 7
- StoreFront
- Operating systems for hosted desktops:
  - Windows Server 2012 R2
  - Windows Server 2012
  - Windows Server 2008 R2
  - Windows 7 (32-bit and 64-bit)
  - Windows 8 (32-bit and 64-bit)
  - Windows 8.1 (32-bit and 64-bit)
- Operating systems for client:
  - Windows XP SP3 (32-bit)
  - Windows 7 (32-bit and 64-bit)
  - Windows 8 (32-bit and 64-bit)
  - Windows 8.1 (32-bit and 64-bit)
- Web Browsers (only the following are supported):
  - Internet Explorer 8, 9, and 10
  - Mozilla Firefox 3.5 through 21.0.
  - Google Chrome 10
  - Citrix Receiver 4.0

Limitations

- Local App Access is only designed for full-screen, virtual desktops spanning all monitors as follows:
  - User experience could be confusing if Local App Access is used with a virtual desktop that runs in windowed mode or does not cover all monitors.
  - For users with multi-monitors, if one monitor is maximized, it becomes the default desktop for all applications launched in that session, even if the subsequent applications typically launch on the other monitor.
  - The feature is designed for use with one VDA; there is no integration with multiple, concurrent VDAs.

- Some applications have unexpected behavior, which could impact users:
  - Users might be confused with drive letters, such as local C: rather than virtual desktop C: drive.
  - Printers available in the virtual desktop are not available to local applications.
  - Applications that require elevated permissions cannot be launched as client-hosted applications.
  - No special handling for single-instance apps (such as Windows Media Player).
  - Local applications appear with the Windows theme of the local machine.
  - Full-screen applications are not supported. This includes applications that open to the full screen, such as PowerPoint slide shows, or photo viewers that cover the entire desktop.
  - Local App Access copies the properties of the local application, such as the shortcuts present on client's desktop and the Start menu on the VDA. However, it does not copy other properties, such as shortcut keys and read-only attributes.
  - Applications that do customize the manipulation of the order of overlapping windows can have unpredictable results.
For example, some windows might be hidden.

- Shortcuts are not supported, including My Computer, Recycle Bin, Control Panel, Network Drive shortcuts, and folder shortcuts.
- The following file types and files are not supported: custom file types, files with no associated programs, zip files, and hidden files.
- Taskbar grouping is not supported for mixed 32-bit and 64-bit client-hosted applications or VDA applications, such as grouping of 32-bit local applications with 64-bit VDA applications, and vice versa.
- Applications cannot be launched using COM. For example, if you click an embedded Office document from within an Office application, the process launch cannot be detected, and the local application integration fails.
- URL Redirection supports only explicit URLs (that is, those appearing in the browser's address bar or found using the in-browser navigation, depending on the specific browser).
- The URL Redirection feature only works with desktop sessions and currently does not work with application sessions.
- The local desktop folder in a VDA session does not allow users to create new files.
Provide access to local applications

Jun 11, 2013

To access local applications, you must enable the Local App Access feature, which is not enabled by default in the hosting and client environment.

Enable this feature by following the procedures in

— To enable Local App Access using policy settings
and
— To enable Local App in Receiver

After completing these procedures, the Local App Access feature is enabled and all the client's shortcuts appear in the desktop on the VDA.

Use this procedure if you want to provide access to only Published Applications.

1. On the machine running Studio, open the Registry Editor and browse to the following path:
   HKLM\Software\Wow6432Node\Citrix\DesktopStudio.
2. Add the following registry entry (of type REG_DWORD): ClientHostedAppsEnabled and add a value of 1 to enable Local App Access. (Using 0 disables Local App Access).
3. Restart the machine, and then restart Studio to implement the changes.
4. Publish the Local App Access applications as described in
   — To publish local applications in Studio

5. Enable Local App Access in Receiver as described in
   — To enable Local App Access in Receiver

6. Set the Allow Local App access policy to Enabled as described in
   — To enable Local App Access using policy settings

1. In Studio, select the Delivery Groups node in the left pane, then the Applications tab in the main pane. Click Create Client-hosted application in the Actions pane. The Create Client-hosted application wizard appears.
2. On the Delivery Groups page, select the Delivery Groups to access the local application.
   Note: Local App Access is only available for desktop Delivery Groups. It is not available for application Delivery Groups.
3. On the Location page, enter the full executable path of the application as present on the user's local machine in the Program Executable field.
   Note: Make sure that this path is correct. Otherwise, the application is not visible to users.
4. On the Users page, add the users who access the local application being published.
   Note: Choose either individual users or a group from your Active Directory configuration.
5. On the Shortcut page, select whether the shortcut to the local application on the virtual desktop is visible on the Start menu, the desktop, or both.
6. On the Content redirection page, specify the File Type Association (FTA) of the application. By default, all the file types associated with the application can be used by selecting Get the file types from user's computer at run time or file types can be provided explicitly by choosing Set the file types explicitly and then adding specific extensions. For example, by default, Microsoft Word associates the Word-related extensions (such as .doc, .docx, .rtf). Selecting Get the file types from user's computer at run time specifies FTA as all extensions. However, if you want to limit the file types associated with the application (For example, only .doc), then select Set the files types explicitly to make those specifications.

7. On the Name page, accept the default values.
8. On the Summary page, review the settings and click Finish.

Follow the procedures in Enable Local App Access on client machines running Receiver to complete the Local App Access feature configuration.

Use Local App Access policy for delivery groups by setting the policy to Enabled. With this setting, the VDA lets the client decide whether administrator-published apps and Local App Access shortcuts are enabled in the session. When set to Disabled, both administrator-published apps and Local App Access shortcuts do not work for the VDA. This policy applies to the entire machine as well as for the URL redirection policy. For more information, see Local App Access policy settings.

Configure the URL Redirection feature

You can optionally use policy settings instead of the Windows Registry to set up the URL Redirection feature by configuring the multi strings lists URL White List and URL Black List policies in Studio. For detailed URL redirection configuration procedures, see Use URL Redirection to launch local applications.

For more information about this using policies, see Local App Access policy settings.

You can configure the expected behavior of local applications when a user logs off or disconnects from their virtual desktop.

1. On the hosted desktop, open the Registry Editor and browse to the path:
   HKEY_LOCAL_MACHINE\SOFTWARE\Citrix\Client Hosted Apps\Policies\Session State.
2. Add the registry entry Terminate of type REG_DWORD, and set the value as follows:
   - 1 — Local applications continue to run when a user logs off or disconnects from the virtual desktop.
   - 3 — Local applications close when a user logs off or disconnects from the virtual desktop.

Note: If the value is set to 1, the local applications are reintegrated from the disconnected session upon reconnect to the virtual desktop if they are still available in the local environment.
Feature interaction with Microsoft Windows versions

Jun 11, 2013

Local App Access behaves differently with Microsoft Windows 8 and Windows Server 2012 when compared to Windows 7 and Windows Server 2008 R2. The Local App Access interaction with Windows includes the following behaviors.

- **Windows 8 and Windows Server 2012 short cut behavior**
  - Windows store applications installed on the client are not enumerated as part of Local App Access shortcuts.
  - Image and video files are usually opened by default using Windows store applications. However, Local App Access enumerates the Windows store applications and opens shortcuts with desktop applications.

- **Local Programs**
  - For Windows 7, the folder is available in the Start menu.
  - For Windows 8, Local Programs is available only when the user chooses All Apps as a category from the Start screen.
  - Not all subfolders are displayed in Local Programs.

- **Windows 8 graphics features for applications**
  - Desktop applications are restricted to the desktop area and are covered by the Start screen and Windows 8 style applications.
  - Local App Access applications do not behave like desktop applications in multi-monitor mode. In multi-monitor mode, the Start screen and the desktop display on different monitors.

- **Windows 8 and Local App Access URL Redirection**
  - **Windows 8** — Because Windows 8 Internet Explorer has no add-ons enabled, use desktop Internet Explorer to enable URL Redirection.
  - **Windows Server 2012** — In Windows Server 2012, Internet Explorer disables add-ons by default due to enhanced security configuration. To implement URL Redirection:
    1. Disable Internet Explorer enhanced configuration.
    2. Reset Internet Explorer options and restart to ensure that add-ons are enabled for standard users.

Multiple instances of a locally-running application behave according to the taskbar settings established for the virtual desktop. However, some shortcuts have the following limitations:

- Shortcuts to locally-running applications are not grouped with running instances of those applications. They are also not grouped with running instances of hosted applications or pinned shortcuts to hosted applications.
- Users can only close windows of locally-running applications from the Taskbar. Although users can pin local application windows to the desktop Taskbar and Start menu, the applications might not launch consistently when using these shortcuts.
Enable Local App Access on client machines running Receiver

Sep 12, 2013

To use Local App Access feature with Citrix Receiver, you must:

- Install Receiver on the local client machine.
- Follow these procedures to complete the Local App Access feature configuration:
  - To enable Local App Access during installation
  - To enable the Local App Access template using the Group Policy editor
- Set the Allow local app access policy setting to Enabled as described in
  — To enable Local App Access in Studio

1. Using the Windows command prompt, change to the directory that contains Citrix Receiver obtained from the Citrix.com download page.
2. Run the Citrix Receiver command to install Citrix Receiver with the Allow_CLIENTHOSTEDAPPSURL flag set as shown in the following examples:
   CitrixReceiver.exe /ALLOW_CLIENTHOSTEDAPPSURL=1
   CitrixReceiverWeb.exe /ALLOW_CLIENTHOSTEDAPPSURL=1
   This enables Local App Access along with URL Redirection and also registers the add-ons required for URL redirection.

1. In the Start menu, open Local Group Policy Editor using gpedit.msc command in the Run field or search for Edit group policy.
2. Add the icaclient.adm template located in Receiver Configuration folder (Usually located in C:\Program Files (x86)\Citrix\Online Plugin\Configuration) to the Local Group Policy Editor by selecting Computer Configuration.
3. Right-click Administrative Templates and choose Add/Remove Templates > Add.
4. Once the icaclient.adm template is successfully added, expand Administrative Templates > Classic Administrative Templates (ADM) > Citrix Components > Citrix Receiver > User experience.
   Note: The icaclient.adm template is also available on the User Configuration once it is added to the Computer Configuration.
5. Choose Local App Access settings.
6. Select Enabled and then choose Allow URL Redirection to use the URL redirection feature. For URL redirection, separately register browser add-ons using the command line.

When you enable the Local App Access setting, the VDA lets the client decide whether administrator-published or Local App Access shortcuts are enabled in the session. This policy (as well as for the URL redirection policy) applies to the entire machine. For more information, see Local App Access policy settings.

1. In Studio, select Policy > Edit Policy.
2. Select Allow local app access > Select.
3. Select Allowed > OK.
4. Click Next and then click Finished.
Use URL Redirection to launch local applications

Jul 25, 2013

In XenDesktop, hosted desktop sessions use URL redirection to launch Local Access applications. URL redirection makes the application available under more than one URL address. It launches a local browser (based on your browser's URL blacklist) by clicking embedded links within a browser in a desktop session. If you navigate to a URL that is not present in the blacklist, the URL is opened in the desktop session again.

In addition to URL redirection, you can also use File Type Association (FTA) redirection. FTA launches local applications when a file is encountered in the session. If the local app is launched, it must have access to the file to open it. Therefore, you can only open files that reside on network shares or on client drives (using CDM) using local apps.

For example, when opening the file `\client\C:\users\<username>\desktop\hugefile.pdf`, if a PDF reader is a local app, then the file opens using that PDF reader. Because the local app can access the file directly, there is no network transfer of the file through ICA to open this file.

Note: These features only work for desktop sessions and do not work for application sessions. The only redirection feature you can use for application sessions is Host-to-client content redirection, which is a type of Server FTA. This FTA redirects certain protocols to the client, such as http, https, rtsp, or mms. For example, if you only open embedded links with http, the links directly open with the client application. There is no URL blacklist or whitelist support.

When Local App Access is enabled for virtual desktops, URLs that are displayed to users as links from locally-running applications, from user-hosted applications, or as shortcuts on the desktop are redirected in one of the following ways:

- From the user's computer to the hosted desktop
- From the XenDesktop server to the user's computer
- Rendered in the environment in which they are launched (not redirected)

URL Redirection provides URL matching functionality and based on a predefined list, the URL is selectively launched on the endpoint or the VDA browser. Use URL redirection for end users with a virtual desktop as their primary work spaces.

To specify the redirection path of content from specific Web sites, configure the URL Whitelist and URL Blacklist on the Delivery Agent. These lists consist of multi-string registry keys that specify the policy for URL redirection as described in Local App Access policy settings.

Although all the URLs can be rendered on the VDA itself, there are some exceptions:

- **Geo/Locale information** — Web sites that require locale information, such as msn.com or news.google.com (opens a country specific page based on the Geo). For example, if the VDA is provisioned from a data center in the UK and the client is connecting from India, the user expects to see in.msn.com but instead sees uk.msn.com.
- **Multimedia content** — Web sites containing rich media content, when rendered on the client device, give the end users a native experience and also save bandwidth even in high latency networks. Although there is Flash redirection feature, this complements by redirecting sites with other media types such as Silverlight. This is in a very secure environment. That is, the URLs that are approved by the administrator are run on the client while the rest of the URLs are redirected to the VDA.

By default, URL redirection is disabled on user devices when you install Receiver. You can enable it on the command line
during installation. To do so, you must:

- **Have Administrator rights**
- **As Administrator, install Receiver for All Users on a machine**
  
  Note: When you install as Administrator, the installation location is `C:\Program Files\Citrix\ICA Client`. Check the installation location to make sure all users can access Receiver.
- **For deployments using the Receiver Standard package, enter:**
  
  `CitrixReceiver.exe ALLOW_CLIENTHOSTEDAPPSURL=1`

  This installs and registers the necessary browser add-ons, and also enables necessary client lockdown settings for enabling LAA, including the URL redirection feature.

The URL redirection feature uses add-ons for Internet Explorer, Google Chrome, and Mozilla Firefox browsers. This feature is installed with Receiver 4.0, and you can also register add-ons using the following commands:

- **Internet Explorer**
  
  `<Client_Installation_Folder>\redirector.exe /regIE`

- **Firefox**
  
  `<Client_Installation_Folder>\redirector.exe /regFF`

- **Chrome**
  
  `<Client_Installation_Folder>\redirector.exe /regChrome`

- **All browsers**
  
  `<Client_Installation_Folder>\redirector.exe /regAll`

  For example, to register IE add-ons on Receiver, enter:

  `C:\Program Files\Citrix\ICA Client\redirector.exe/regIE`

To unregister add-ons

- **Internet Explorer**
  
  `<Client_Installation_Folder>\redirector.exe /unregIE`

- **Firefox**
  
  `<Client_Installation_Folder>\redirector.exe /unregFF`

- **Chrome**
  
  `<Client_Installation_Folder>\redirector.exe /unregChrome`

- **All browsers**
  
  `<Client_Installation_Folder>\redirector.exe /unregAll`

Register Add-ons on hosted desktops using the following commands.

- **Internet Explorer**
  
  `<VDA_Installation_Folder>\VDARedirector.exe /regIE`

- **Firefox**
  
  `<VDA_Installation_Folder>\VDARedirector.exe /regFF`

- **Chrome**
  
  `<VDA_Installation_Folder>\VDARedirector.exe /regChrome`

- **All browsers**
  
  `<VDA_Installation_Folder>\VDARedirector.exe /regAll`
**Examples**

Register IE add-ons on Desktop OS VDA (Windows 7 or Windows 8):

C:\Program Files\Citrix\ICAService\VDARedirector.exe /regIE

Register IE add-ons on a Server OS VDA (Windows Server 2008 R2 or Windows Server 2012):

C:\Program Files (x86)\Citrix\System32\VDARedirector.exe/regIE

To unregister Add-ons on hosted desktops

- Internet Explorer
  - `<VDA_Installation_Folder>\VDARedirector.exe /unregIE`
- Firefox
  - `<VDA_Installation_Folder>\VDARedirector.exe /unregFF`
- Chrome
  - `<VDA_Installation_Folder>\VDARedirector.exe /unregChrome`
- All browsers
  - `<VDA_Installation_Folder>\VDARedirector.exe /unregAl`

<table>
<thead>
<tr>
<th>Description</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>By default, Internet Explorer redirects the URL entered. If the URL is not in the blacklist but is redirected to another URL by browser or website, the final URL is not redirected even if it is present in the blacklist.</td>
<td>For URL redirection to work correctly, enable the add-on when prompted by the browser. If the add-ons using internet options or the add-ons in the prompt are disabled, URL redirection does not work correctly.</td>
</tr>
<tr>
<td>Firefox add-ons always redirect the URLs.</td>
<td>When an add-on is installed by a user or with an installer, Firefox prompts you to allow/stop the add-on installation in new tab page. You must allow the add-on for the feature to work.</td>
</tr>
<tr>
<td>Chrome add-on always redirects the final URL that is navigated and not entered URLs.</td>
<td>The extensions have been installed externally. If you disable the extension, the URL redirection feature does not work in Chrome. If the URL redirection is required in Incognito mode, allow the extension to run in Incognito mode by selecting this option.</td>
</tr>
</tbody>
</table>
High definition user experience with HDX

Nov 05, 2013

Citrix HDX includes a broad set of technologies that provide a high-definition user experience.

<table>
<thead>
<tr>
<th>At the device:</th>
<th>HDX leverages the computing capacity of user devices to enhance and optimize the user experience. HDX MediaStream technology ensures users receive a smooth, seamless experience with multimedia content in their virtual desktops or applications. Workspace control enables users to pause virtual desktops and applications and resume working from a different device, at exactly the point where they left off.</th>
</tr>
</thead>
<tbody>
<tr>
<td>On the network:</td>
<td>HDX incorporates advanced optimization and acceleration capabilities to deliver the best performance over any network, including low-bandwidth and high-latency WAN connections. HDX features adapt to changes in the environment, balancing performance and bandwidth by applying the best technologies for each unique user scenario, whether the desktop or application is accessed locally on the corporate network, or remotely from outside the corporate firewall.</td>
</tr>
<tr>
<td>In the datacenter:</td>
<td>HDX leverages the processing power and scalability of servers to deliver advanced graphical performance, regardless of the capabilities of the client device. Compressed multimedia information is sent directly to the user device in its native format. HDX channel monitoring, provided by Citrix Director, shows the status of connected HDX channels on user devices. HDX Insight, the integration of EdgeSight Network Inspector and EdgeSight Performance management with Director, captures data about ICA traffic and provides a Dashboard view of real-time and historical details such as client-side and server-side ICA session latency, bandwidth usage of ICA channels, and the ICA Round Trip Time value of each session.</td>
</tr>
</tbody>
</table>

For support and requirements information, see [HDX system requirements](https://docs.citrix.com).

Citrix policy settings that provide the best out-of-the-box experience for the majority of use cases are enabled by default. The topics in this section describe how to further optimize the user experience, improve server scalability, or reduce bandwidth requirements. For information about working with Citrix policies, refer to [Manage Citrix policies](https://docs.citrix.com).

Note: Except where otherwise noted in this section, HDX features are available for Windows Server OS, Windows Desktop OS, and Remote PC Access desktops.

1. Log on to your virtual desktop.
2. To experience how HDX delivers rich video content to virtual desktops, visit a website containing high-definition videos, such as [http://www.microsoft.com/silverlight/iis-smooth-streaming/demo/](http://www.microsoft.com/silverlight/iis-smooth-streaming/demo/), and view a video.
3. To experience Flash Redirection:
   2. On the Desktop Viewer toolbar, click Preferences. In the Desktop Viewer Preferences dialog box, click the Flash tab and select Optimize content.
   3. To experience how Flash Redirection accelerates the delivery of Flash multimedia content to virtual desktops, visit a
website containing Flash videos, such as http://www.youtube.com/, and view a video on your desktop. Flash Redirection is designed to be seamless so that users will not know when it is running. However, you can check to see whether Flash Redirection is being used by looking for a block of color that appears momentarily before the Flash player starts.

4. To experience HDX delivery of high definition audio to virtual desktops:
   1. Configure your Citrix client for maximum audio quality. Refer to the Receiver documentation for details.
   2. Install a digital audio player (such as iTunes, available from http://www.apple.com/itunes/download/) on your desktop and play some music files.
HDX system requirements

Jul 09, 2014

User’s Windows device or thin client:

- DirectX 9 support
- Pixel Shader 2.0 supported in hardware
- 32 bits per pixel
- 1.5 GHz 32-bit or 64-bit processor
- 1 GB RAM
- 128 MB of video memory on the graphics card or integrated graphics processor

Note: HDX queries the Windows device to verify that it has the required GPU capabilities and automatically reverts to server-side desktop composition if it does not. Windows devices with the required GPU capabilities but that do not meet the processor speed or RAM specifications should be listed in the GPO group for devices excluded from Desktop Composition Redirection.

Bandwidth:

- Minimum available: 1.5 Mbps
- Recommended: 5 Mbps

Note: The minimum available and recommended values incorporate end-to-end latency.

- Supported clients for Windows Media client-side content fetching, Windows Media redirection, and real-time Windows Media multimedia transcoding:
  - Receiver for Windows
  - Receiver for iOS
  - Receiver for Linux

- To use Windows Media client-side content fetching on Windows 8 devices, set the Citrix Multimedia Redirector as a default program in Control Panel > Programs > Default Programs > Set your default programs, select Citrix Multimedia Redirector and click either Set this program as default or Choose defaults for this program.

- For GPU transcoding, a NVIDIA CUDA-enabled GPU with Compute Capability 1.1 or higher. NVIDIA maintains a list at http://developer.nvidia.com/cuda/cuda-gpus.

User device:

- To use a Virtual Delivery Agent (VDA), enable a network connection between the user’s Windows device and the VDA.

  Supported clients:
  - Receiver for Windows (for second generation Flash Redirection features)
  - Receiver for Linux (for second generation Flash Redirection features)
  - Citrix Online plug-in 12.1 (for legacy Flash Redirection features only)

  Supported Adobe Flash Player:
  - For Windows: Second generation Flash Redirection features require Adobe Flash Player - Other Browsers, sometimes referred to as an NPAPI (Netscape Plugin Application Programming Interface) Flash Player
● For Linux: Second generation Flash Redirection features require Adobe Flash Player for other Linux or Adobe Flash Player for Ubuntu.

● Legacy Flash Redirection features require Adobe Flash Player for Windows Internet Explorer (sometimes referred to as an ActiveX player)

Note: The major version number of the Flash Player on the user device must be greater than or equal to the major version number of the Flash Player on the server. If an earlier version of the Flash Player is installed on the user device, or if the Flash Player cannot be installed on the user device, Flash content is rendered on the server.

Server or workstation running Virtual Delivery Agents:

● Adobe Flash Player for Windows Internet Explorer (the ActiveX player)

● Internet Explorer versions 7, 8, 9, or 10 (in non-Metro mode).

Note: Flash redirection requires Internet Explorer on the server; with other browsers, Flash content is rendered on the server.

● Protected mode disabled: From the Tools menu in Internet Explorer, select Internet Options. On the Security tab, clear the Enable Protected Mode check box. Restart Internet Explorer for the change to take effect.

Servers and user devices must meet the minimum hardware requirements for the installed operating system. For more information on the hardware used by Citrix to test HDX 3D Pro, see http://support.citrix.com/article/CTX131385.

Host:

The Virtual Delivery Agent for HDX 3D Pro is supported for installation on the following versions of Windows:

● Windows Server 2012 R2

● Windows Server 2012

● Windows Server 2008 R2

● Windows 8.1 64-bit and 32-bit editions

● Windows 8 64-bit and 32-bit editions

● Windows 7 64-bit editions with Service Pack 1

● Windows 7 32-bit editions with Service Pack 1

● Windows XP Professional 64-bit edition with Service Pack 2

● Windows XP Professional 32-bit edition with Service Pack 3

The Virtual Delivery Agent for HDX 3D Pro can be used with XenApp 7.5, XenDesktop 7.5, XenDesktop 7.1, XenDesktop 7, or XenDesktop 5.6.

HDX 3D Pro can be used to deliver any application that is compatible with the supported host operating systems, but is particularly suitable for use with DirectX and OpenGL-driven applications and with rich media such as video.

The computer hosting the application can be either a physical machine or a virtual machine with GPU Passthrough or Virtual GPU (vGPU):

● The GPU Passthrough feature is available with Citrix XenServer on Citrix Downloads. GPU Passthrough is also available with VMware vSphere and VMware ESX, where it is referred to as virtual Direct Graphics Acceleration (vDGA).

● The vGPU feature is available with Citrix XenServer on Citrix Virtual GPU Solution.

Citrix recommends that the specification of the host computer include at least 4 GB of RAM and four virtual CPUs with a clock speed of 2.3 GHz or higher.

Graphical Processing Unit (GPU):
For CPU-based compression, including lossless compression, HDX 3D Pro supports any display adapter on the host computer that is compatible with the application that you are delivering.

For optimized GPU frame buffer access using the NVIDIA GRID API, HDX 3D Pro requires NVIDIA Quadro cards with the latest NVIDIA drivers. The NVIDIA GRID delivers a high frame rate, resulting in a highly interactive user experience.

For vGPU using XenServer, HDX 3D Pro requirements include NVIDIA GRID K1 and K2 cards. For all vGPU requirements, see Citrix Virtual GPU Solution.

User device:

To access desktops or applications delivered with HDX 3D Pro, users must install Citrix Receiver. For more information on Receiver system requirements, see Receiver and Plug-ins.

HDX 3D Pro supports all monitor resolutions that are supported by the GPU on the host computer. However, for optimum performance with the minimum recommended user device and GPU specifications, Citrix recommends maximum monitor resolutions for users' devices of 1920 x 1200 pixels for LAN connections and 1280 x 1024 pixels for WAN connections.

Citrix recommends that user devices include at least 1 GB of RAM and a CPU with a clock speed of 1.6 GHz or higher. Use of the default deep compression codec, which is required on low-bandwidth connections, requires a more powerful CPU unless the decoding is done in hardware. For optimum performance, Citrix recommends that users' devices are equipped with at least 2 GB of RAM and a dual-core CPU with a clock speed of 3 GHz or higher.

For multi-monitor access, Citrix recommends user devices equipped with quad-core CPUs.

User devices do not need a dedicated GPU to access desktops or applications delivered with HDX 3D Pro.

Supported client:

- Receiver for Windows
- Receiver for Linux 13

Supported clients:

- Receiver for Windows
- Receiver for Mac
- Receiver for Linux

Supported video conferencing applications:

- Citrix GoToMeeting HDFaces
- Adobe Connect
- Cisco WebEx
- IBM Sametime
- Microsoft Lync 2010
- Microsoft Office Communicator
Google+ Hangouts (compatible with VDAs running Windows 7)

Skype 6.7. To use Skype 6.7 from a Windows client, you need to make registry edits on both the client and server:

- From the client, locate the registry key: HKEY_CURRENT_USER\Software\Citrix\HdxRealTime
  - Name: DefaultHeight
    Type: REG_DWORD
    Data: 240
  - Name: DefaultWidth
    Type: REG_DWORD
    Data: 320

- From the server, locate the registry key: HKEY_LOCAL_MACHINE\SOFTWARE\Citrix\Vd3d\Compatibility
  - Name: skype.exe
    Type: REG_DWORD
    Data: Set to 0

Other user device requirements:

- Appropriate hardware to produce sound
- DirectShow-compatible webcam (Use the webcam default settings.)
  Note: Use of webcams that are hardware encoding capable reduces client-side CPU usage.
- Webcam drivers, obtained from the camera manufacturer if possible, installed
Optimize graphics and multimedia delivery

Aug 05, 2014

XenApp and XenDesktop provide a superior graphics and video experience for most users by default, with no configuration required.

- HDX automatically selects the best delivery method based on the client, platform, application, and network bandwidth, and then self-tunes based on changing conditions.
- HDX optimizes the performance of 2D and 3D graphics and video.
- HDX delivers a Windows Aero experience to virtual desktop users on any client.
- HDX enables user devices to stream multimedia files directly from the source provider on the Internet or Intranet, rather than through the host server. If the requirements for this client-side content fetching are not met, media delivery falls back to Windows Media redirection to play media run-time files on user devices rather than the host server. In most cases, no adjustments to the Windows Media feature policies are needed.

By default, XenDesktop deliver a highly responsive Windows Aero or Windows 8 desktop experience to virtual desktops accessed from Windows user devices. To do that, XenDesktop leverage the graphics processing unit (GPU) or integrated graphics processor (IGP) on supported Windows user devices for local DirectX graphics rendering. This feature, referred to as desktop composition redirection, maintains high scalability on the server.

For detailed information, see — What to do with all these choices in http://blogs.citrix.com/2013/11/06/go-supersonic-with-xendesktop-7-x-bandwidth-supercodescs/#Choices.

Important: For non-Windows clients or when desktop composition redirection is not available, HDX processes graphics on the server, compresses the bitmap image, and sends it over the network to the client. Aero themes are supported, but are not required.

Users of the latest Citrix Receiver releases will notice improved performance for server-rendered video.

To reduce the bandwidth required in user sessions, consider adjustments to the following policy settings. Keep in mind that adjustments to these settings can reduce the quality of the user experience.

- Desktop Composition Redirection – Applies only to Windows Desktop OS machines accessed from Windows user devices and applies only to the composition of the Windows desktop. Application windows are rendered on the server unless the Citrix policy setting Allow local app access is Allowed.
- Desktop Composition Redirection graphics quality – Uses high-quality graphics for desktop composition unless seamless applications or Local App Access are enabled. To reduce bandwidth requirements, lower the graphics quality.
- Dynamic windows preview – Controls the display of seamless windows to provide the following preview options to virtual desktop users.

<table>
<thead>
<tr>
<th>Windows Aero preview option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Taskbar Preview</td>
<td>When the user hovers over a window's taskbar icon, an image of that window appears above the taskbar.</td>
</tr>
<tr>
<td>Windows Peek</td>
<td>When the user hovers over a taskbar preview image, a full-sized image of the window</td>
</tr>
</tbody>
</table>
When the user presses ALT+TAB, small preview icons are shown for each open window.

When the user presses TAB+Windows logo key, large images of the open windows cascade across the screen.

To reduce bandwidth requirements, disable this policy setting.

The following Visual Display policy settings control the quality of images sent from virtual desktops to user devices.

- **Visual quality** – Controls the visual quality of images displayed on the user device.
  - Medium – Medium visual quality offers the best performance and bandwidth efficiency in most use cases. This is the default setting.
  - High – The High visual quality setting is recommended if you require visually lossless image quality.
  - Always lossless – The Always lossless setting ensures that imaging is always pixel-perfect. This setting is necessary in some specialized use cases.
  - Build to lossless – The Build to lossless setting sends display data using lossy compression while images are in transit and returns to lossless image quality when the transfer stops. This setting improves performance over bandwidth-constrained network connections.

- **Target frame rate** – Specifies the maximum number of frames per second that are sent from the virtual desktop to the user device. The default is 30 frames per second. In many circumstances, you can improve the user experience by specifying a higher value.

- **Display memory limit** – Specifies the maximum video buffer size for the session in kilobytes. The default is 65536 KB. For connections requiring more color depth and higher resolution, increase the limit. Calculate the maximum memory required using this equation:
  \[
  \text{memory required} = \left(\frac{\text{color depth in bits per pixel}}{8}\right) \times \text{vertical resolution in pixels} \times \text{horizontal resolution in pixels}
  \]

  For example, if the color depth is 32, the vertical resolution is 600, and the horizontal resolution is 800, then the maximum memory required is \(\left(\frac{32 \text{ bpp}}{8}\right) \times 600 \text{ pixels} \times 800 \text{ pixels} = 1920000 \text{ bytes}\), so you would set the Display memory limit to 1920 KB.

  Color depths other than 32-bit are available only if the Legacy graphics mode policy is enabled.

To improve bandwidth efficiency and latency tolerance during video conferencing in a session, HDX webcam video compression is used for webcams by default. HDX webcam video compression streams webcam traffic over a dedicated multimedia virtual channel. HDX webcam video compression uses significantly less bandwidth compared to the isochronous HDX Plug-n-Play support and works well over WAN connections.

However, Receiver users can override the default behavior by choosing the Desktop Viewer Mic & Webcam setting Don’t use my microphone or webcam. To prevent users from switching from HDX webcam video compression, disable USB device redirection through the policy settings under ICA > USB Devices.
Note: HDX webcam video compression is enabled by default on Receiver for Windows but must be configured on Receiver for Linux. For more information, refer to the Receiver documentation in eDocs.
HDX webcam video compression requires that the following policy settings are enabled. All of these settings are enabled by default.

- Client audio redirection
- Client microphone redirection
- Multimedia conferencing
- Windows Media Redirection

**To force software compression over low bandwidth networks**

If a webcam supports H.264 hardware encoding, HDX video compression uses the hardware encoding by default. Hardware encoding uses additional bandwidth and is not suitable for a low bandwidth network.

Caution: Editing the Registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

To force software compression, add the following DWORD key value to the registry key HKCU\Software\Citrix\HdxRealTime: DeepCompress_ForceSWEncode=1.

For deployments where server scalability is of greater concern than user experience, you can use the legacy graphics system by adding the Legacy graphics mode setting to a policy and configuring the individual legacy graphics policy settings. Use of the legacy graphics system particularly impacts the user experience over WAN and mobile connections.
Configure Flash Redirection

Jun 13, 2013

Flash Redirection offloads the processing of most Adobe Flash content (including animations, videos, and applications) to users’ LAN- and WAN-connected Windows devices. By moving the processing to the user device rather than using server resources, Flash Redirection reduces server and network load. This results in greater scalability while ensuring a high definition user experience. Configuring Flash Redirection requires both server-side and client-side settings.

Caution: Flash Redirection involves significant interaction between the user device and server components. This feature should be used only in environments where security separation between the user device and server is not required. Additionally, user devices should be configured to use this feature only with trusted servers. Because Flash Redirection requires the Flash Player to be installed on the user device, this feature should be enabled only if the Flash Player itself is secured.

The legacy and second generation versions of Flash Redirection are independent solutions and run in separate virtual channels.

- Legacy Flash Redirection features are supported on the client side only. If an earlier version of the Flash Player is installed on the user device, or if the Flash Player cannot be installed, Flash content renders on the server.
- Second generation Flash Redirection is supported on both clients and servers. If the client supports second generation Flash Redirection, Flash content renders on the client. Second generation Flash Redirection features include:
  - Support for user connections over WAN.
  - Intelligent Fallback, which determines on a per-instance basis when it is more efficient to render the Flash content on the server.
  - Flash URL Compatibility List, which controls whether specific URLs should be rendered on the client, rendered on the server, or blocked.

For the latest updates to HDX Flash compatibility, refer to CTX136588.

Flash Redirection uses Windows event logging on the server to log Flash events. The event log indicates whether Flash Redirection is being used and provides details about issues. The following are common to all events logged by Flash Redirection:

- Flash Redirection reports events to the Application log.
- On Windows 8 and Windows 7 systems, a Flash Redirection-specific log appears in the Applications and Services Logs node.
- The Source value is Flash.
- The Category value is None.
Configure Flash Redirection on the server

Aug 11, 2014

To configure Flash Redirection on the server, use the following Citrix Policy settings:

- Flash default behavior
- Flash intelligent fallback
- Flash server-side content fetching URL list
- Flash URL compatibility list
- Flash background color list

The Citrix policy setting Flash default behavior establishes the default behavior of Flash acceleration. By default, Flash Redirection is enabled.

To override this default behavior for individual web pages and Flash instances, use the Flash URL compatibility list setting.

Three options are available:

<table>
<thead>
<tr>
<th>Option</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block Flash player</td>
<td>Flash Redirection and server-side rendering are not used. The user cannot view any Flash content.</td>
</tr>
<tr>
<td>Disable Flash acceleration</td>
<td>Flash Redirection is not used. The user can view server-side rendered Flash content if a version of Adobe Flash Player for Windows Internet Explorer compatible with the content is installed on the server.</td>
</tr>
<tr>
<td>Enable Flash acceleration</td>
<td>Flash Redirection is used; this is the default. Important: This option requires that the Enable HDX MediaStream Flash Redirection on the user device Group Policy Objects policy setting is enabled on the user device.</td>
</tr>
</tbody>
</table>

Flash intelligent fallback is enabled by default, to detect instances of small Flash movies (such as those frequently used to play advertisements) and to render them on the server instead of redirecting them for rendering on the user device.

Flash intelligent fallback does not cause any interruption or failure in the loading of the web page or the Flash application.

To redirect all instances of Flash content for rendering on the user device, disable the Flash intelligent fallback policy setting.

By default, Flash Redirection downloads Flash content to the user device, where it is played. The Flash server-side content fetching URL list policy setting allows you to specify websites whose Flash content can instead be downloaded to the server and then transferred to the user device for rendering. This setting works with the Enable server-side content fetching setting on the user device and is primarily intended for use with Intranet sites and internal Flash applications. It also works with most Internet sites and can be used when the user device does not have direct access to the Internet (for...
example, when the XenApp or XenDesktop server provides that connection).

Note: Server-side content fetching does not support Flash applications using Real Time Messaging Protocols (RTMP); instead, server-side rendering is used, which supports HTTP and HTTPS.

To create the list of allowed URLs, add the Flash server-side content fetching URL list setting to a policy and click New to add URLs to the list. When configuring the Flash server-side content fetching URL list setting:

- Add the URL of the Flash application, not the top-level .html page that instantiates the Flash Player.
- Use an asterisk character at the beginning or end of the URL as a wildcard to expand your list.
- Use a trailing wildcard to allow all child URLs, for example http://www.sitetoallow.com/*.  
- The prefixes http:// or https:// are not required, but are used if present.

Important: On the user device, be sure to enable the Enable server-side content fetching policy setting.

The Flash URL compatibility list policy setting lets you specify whether Flash content from listed websites is:

- Rendered on the user device
- Rendered on the server
- Blocked from rendering

When configuring the Flash URL compatibility list setting:

- Prioritize the list, placing the most important URLs, actions, and rendering locations at the top.
- Add sites containing Flash content that does not render correctly on the user device, specifying the Render on Server or Block option.
- Use an asterisk character at the beginning or end of the URL as a wildcard to expand your list.
- Use a trailing wildcard to refer to all child URLs, for example http://www.sitetoblock.com/*.
- The prefixes http:// or https:// are not required, but are used if present.

To add the Flash URL compatibility list setting to a policy:
1. Click New to open the Add Flash URL Compatibility list entry dialog box.
2. Select an Action: Render on Client, Render on Server, or Block.
3. In the URL Pattern box, type the URL of the website.
4. Select the Flash instance you want to serve as a trigger:
   - Select Any to have the action occur any time any Flash instance connects with the listed website.
   - Select Specific and specify the Flash player ID to have the action occur only when this specific Flash instance connects with the listed website.

To improve the appearance of the web page when using Flash Redirection, use the Flash background color list policy setting. This enables you to match the colors of web pages and Flash instances.

When configuring the Flash background color list setting:

- For best results, consider using a color not typically used on the web page, such as black.
- Use a trailing wildcard to enable matching in all child URLs, for example, http://www.sitetomatch.com/* FF0000.

When adding the Flash background color list setting to a policy:
1. Click New and type the website URL followed by the appropriate 24-bit Web color hexadecimal number (for example, http://www.sitetomatch.com/ FF0000).
2. Click the Accept button.
Configure Flash Redirection on the user device

Oct 17, 2013
To use Flash Redirection, install Citrix Receiver and the Adobe Flash Player on the user device. No further configuration is required on the user device. However, you can change the default settings using Group Policy Objects as described in this section.

1. Create or select an existing Group Policy Object.
2. Import and add the HDX MediaStream Flash Redirection - Client administrative template (HdxFlash-Client.adm), available in:
   - For 32-bit computers: %Program Files%\Citrix\ICA Client\Configuration\language
   - For 64-bit computers: %Program Files (x86)\Citrix\ICA Client\Configuration\language

   The policy settings appear under Administrative Templates > Classic Administrative Templates (ADM) > HDX MediaStream Flash Redirection - Client.

   Note: For details on creating Group Policy Objects and importing and adding templates, see the Microsoft Active Directory documentation at http://www.microsoft.com

In conjunction with server-side settings, the Enable HDX MediaStream Flash Redirection on the user device policy setting controls whether Adobe Flash content is redirected to the user device for local rendering.

Note: If no configuration is set and Desktop Lock is used, Flash Redirection is enabled on the user device by default. By default, Flash Redirection is enabled and uses intelligent network detection to determine when to play Flash content on the user device. To change when Flash Redirection is used or to disable Flash Redirection on the user device:

1. From the Setting list, select Enable HDX MediaStream Flash Redirection on the user device and click policy setting.
2. Select Not Configured, Enabled (the default), or Disabled.
3. If you selected Enabled, choose an option from the Use HDX MediaStream Flash Redirection list:
   - To use the latest Flash Redirection functionality when the required configuration is present, and revert to server-side rendering when it is not, select Only with Second Generation.
   - To always use Flash Redirection, select Always. Flash content plays on the user device.
   - To never use Flash Redirection, select Never. Flash content plays on the server.
   - To use intelligent network detection to assess the security level of the client-side network to determine when using Flash Redirection is appropriate, select Ask (the default). If the security of the network cannot be determined, the user is asked whether to use Flash Redirection. If the network security level cannot be determined, the user is prompted to choose whether to use Flash Redirection.

   The following illustration indicates how the Flash Redirection is handled for various network types.
Note: Users can override intelligent network detection from the Citrix Receiver - Desktop Viewer Preferences dialog box by selecting Optimize or Don't Optimize in the Flash tab. The choices available vary depending on how Flash Redirection is configured on the user device, as shown in the following illustration.
Synchronization of the client-side HTTP cookies with the server-side is disabled by default. Enable synchronization to download HTTP cookies from the server. These HTTP cookies are then used for client-side content fetching and are available as needed by sites containing Flash content.

Note: Client-side cookies are not replaced during the synchronization; they remain available even if the synchronization policy is later disabled.

1. From the Setting list, select Enable synchronization of the client-side HTTP cookies with the server-side and click policy setting.
2. Select Not Configured, Enabled, or Disabled (the default).

By default, Flash Redirection downloads Adobe Flash content to the user device, where it is played. Enabling server-side content fetching causes the Flash content to download to the server and then be sent to the user device. Unless there is an overriding policy (such as a site blocked through the Flash URL compatibility list policy setting), the Flash content plays on the user device.

Server-side content fetching is frequently used when:
The user device connects to internal sites through Citrix NetScaler Gateway.

The user device does not have direct access to the Internet.

Note: Server-side content fetching does not support Flash applications using Real Time Messaging Protocols (RTMP). Instead, server-side rendering is used for such sites.

Second generation Flash Redirection supports three enabling options for server-side content fetching, as described in the following table. Two of these options include the ability to cache server-side content on the user device; this improves performance because content that is reused is already available on the user device for rendering.

Note: The contents of this cache are stored separately from other HTTP content cached on the user device.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disabled</td>
<td>Disables server-side content fetching, overriding the Flash server-side content fetching URL list setting on the server. Server-side content fetching fallback is also disabled.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Enables server-side content fetching for web pages and Flash applications identified in the Flash server-side content fetching URL list. Server-side content fetching fallback is available, but Flash content is not cached.</td>
</tr>
<tr>
<td>Enabled (persistent caching)</td>
<td>Enables server-side content fetching for web pages and Flash applications identified in the Flash server-side content fetching URL list. Server-side content fetching fallback is available. Content obtained through server-side fetching is cached on the user device and stored from session to session.</td>
</tr>
<tr>
<td>Enabled (temporary caching)</td>
<td>Enables server-side content fetching for web pages and Flash applications identified in the Flash server-side content fetching URL list. Server-side content fetching fallback is available. Content obtained through server-side fetching is cached on the user device and deleted at the end of the session.</td>
</tr>
</tbody>
</table>

Note: With second generation Flash redirection, fallback to server-side content fetching begins automatically when any of the above enabling options is selected and client-side fetching of .swf files fails.

Enabling server-side content fetching requires settings on both the client device and the server:

1. From the Setting list, select Enable server-side content fetching and click policy setting.
2. Select Not Configured, Enabled, or Disabled (the default). If you are enabling this setting, choose an option from the Server-side content fetching state list:
   - Disabled
     Note: This setting is preserved in the Registry.
   - Enabled
   - Enabled (persistent caching)
   - Enabled (temporary caching)
3. On the server, enable the Flash server-side content fetching URL list policy setting and populate it with target URLs.

You can redirect an attempt to obtain Flash content using the URL rewriting rules for client-side content fetching setting, which is a second generation Flash Redirection feature. When configuring this feature, you provide two URL patterns using Perl regular expressions. If the user device attempts to fetch content from a website matching the first pattern (the URL match pattern), it is redirected to the website specified by the second pattern (the rewritten URL format).
You can use this setting to compensate for content delivery networks (CDN). Some websites delivering Flash content use CDN redirection to enable the user to obtain the content from the nearest of a group of servers containing the same content. When using the Flash Redirection client-side content fetching feature, the Flash content is requested from the user device, while the rest of the web page on which the Flash content resides is requested by the server. If CDN is in use, the server request is redirected to the closest server, and the user device request follows to the same location. Note that this may not be the location closest to the user device; depending on distance, there could be a noticeable delay between the loading of the web page and the playing of the Flash content.

1. From the Setting list, select URL rewriting rules for client-side content fetching and click policy setting.
2. Select Not Configured, Enabled, or Disabled. Not Configured is the default; Disabled causes any URL rewriting rules configured in the next step to be ignored.
3. If you selected Enabled, click Show. Using Perl regular expression syntax, type the URL match pattern in the Value name box and the rewritten URL format in the Value box.
HDX 3D Pro

Dec 11, 2013
HDX 3D Pro enables you to deliver desktops and applications that perform best with a graphics processing unit (GPU) for hardware acceleration, including 3D professional graphics applications based on OpenGL and DirectX. (The standard VDA supports GPU acceleration of DirectX only.)

Example 3D professional applications include:

- Computer-aided design, manufacturing, and engineering (CAD/CAM/CAE) applications
- Geographical Information System (GIS) software
- Picture Archiving Communication System (PACS) for medical imaging
- Applications using the latest OpenGL, DirectX, NVidia CUDA, and OpenCL versions
- Computationally-intensive non-graphical applications that use NVIDIA Compute Unified Device Architecture (CUDA) GPUs for parallel computing

HDX 3D Pro provides the best user experience over any bandwidth:

- On wide area network (WAN) connections: Deliver an interactive user experience over WAN connections with bandwidths as low as 1.5 Mbps.
- On local area network (LAN) connections: Deliver a user experience equivalent to that of a local desktop on LAN connections with bandwidths of 100 Mbps.

HDX 3D Pro enables you to replace complex and expensive workstations with much simpler user devices by moving the graphics processing into the data center for centralized management.

Note: The HDX Monitor tool, which replaced the Health Check tool, enables you to validate operation and configuration of HDX visualization technologies and to diagnose and troubleshoot HDX issues. To download the tool and learn more about it, see https://taas.citrix.com/hdx/download/.

HDX 3D Pro provides GPU acceleration for Windows Desktop OS machines and Windows Server OS machines. When used with Citrix XenServer and NVIDIA GRID GPUs, HDX 3D Pro provides Virtual GPU (vGPU) acceleration for Windows Desktop OS machines. For the supported XenServer versions, see Citrix Virtual GPU Solution.
GPU acceleration for Windows Desktop OS

Feb 05, 2016
With HDX 3D Pro you can deliver graphically intensive applications as part of hosted desktops or applications on Desktop OS machines, according to the requirements of your users. HDX 3D Pro supports physical host computers (including desktop, blade, and rack workstations) and XenServer VMs with GPU Passthrough and XenServer VMs with Virtual GPU (vGPU).

The XenServer GPU Passthrough feature enables you to create VMs with exclusive access to dedicated graphics processing hardware. You can install multiple GPUs on the hypervisor and assign VMs to each of these GPUs on a one-to-one basis.

The XenServer vGPU feature enables multiple virtual machines to directly access the graphics processing power of a single GPU.

- **vGPU** — The Virtual Graphical Processing Unit (vGPU) feature enables multiple virtual machines to directly access the graphics processing power of a single physical GPU. You can use hardware-accelerated vGPU access for Windows desktop VDI workloads. The true hardware GPU sharing provides full Windows 7 or Windows 2008 R2 SP1 desktops suitable for users with complex and demanding design requirements. Supported for NVIDIA GRID K1 and K2 cards, the GPU sharing uses the same NVIDIA graphics drivers that are deployed on non-virtualized operating systems.

- **Flash Redirection support on more operating systems** — Flash Redirection is now supported for Windows Server 2012 R2, Windows Server 2012, Windows 8.1, and Windows 8 VDAs. Flash Redirection is enabled by default for those operating systems.

- **Application window positions retained in local sessions** — When a user roams from a local to a remote session, and then roams back to the local session, the application windows are returned to their previous positions on the local session. XenDesktop stores window positions by client name.

- **Adaptive H.264-based deep compression for optimal WAN and wireless performance** - HDX 3D Pro uses CPU-based deep compression as the default compression technique for encoding. This provides optimal compression that dynamically adapts to network conditions. The H.264-based deep compression codec no longer competes with graphics rendering for CUDA cores on the NVIDIA GPU. The deep compression codec runs on the GPU and provides superior bandwidth efficiency over prior HDX 3D Pro releases.

- **Lossless compression option for specialized use cases** - HDX 3D Pro also offers a CPU-based lossless codec to support applications where pixel-perfect graphics are necessary, such as medical imaging. H.264-based deep compression codec provides significantly better bandwidth efficiency than was available in the previous XenDesktop release. Lossless compression is recommended only for specialized use cases because it consumes significantly more network and processing resources.

  When using lossless compression:

  - The lossless indicator, a system tray icon, notifies the user if the screen displayed is a lossy frame or a lossless frame. This helps when the VisualQuality is set to BuildToLossless. The lossless indicator turns green when the frames sent are lossless.
  - The lossless switch enables the user to change to Always Lossless mode anytime within the session. To select or deselect Lossless anytime within a session, right-click the icon or use the shortcut ALT+SHIFT+1.

    For lossy compression: HDX 3D Pro uses the lossless codec for compression regardless of the codec selected through policy.

    For lossy compression: HDX 3D Pro uses the original codec, either the default or the one selected through policy.

    Note: Lossless switch settings are not retained for subsequent sessions. To use lossless codec for every connection, set the Visual quality policy to Always lossless.

- **Multiple monitor support** - For Windows 7 and Windows 8 desktops, HDX 3D Pro supports user devices with up to four monitors. Users have the freedom to arrange their monitors in any configuration and can mix monitors with different resolutions and orientations. The number of monitors is limited by the capabilities of the host computer GPU, the user device, and the available bandwidth.
HDX 3D Pro also provides limited support for dual-monitor access to Windows XP desktops. The XenDesktop 5.6 VDA, included with XenDesktop 7 and XenDesktop 7.1, is required to deliver Windows XP virtual desktops and applications.

- **High resolution monitor support** - HDX 3D Pro supports all monitor resolutions and is only limited by the capabilities of the GPU on the host computer.
- **Dynamic resolution** - You can resize the virtual desktop or application window to any resolution in this release. **Note**: The only supported method to change the resolution is by resizing the VDA session window. Changing resolution from within the VDA session (using Control Panel > Appearance and Personalization > Display > Screen Resolution) is not supported.
- **Support for NVIDIA Kepler architecture** - HDX 3D Pro supports NVIDIA GRID K1 and K2 cards for GPU passthrough and GPU sharing. NVIDIA GRID vGPU enables multiple VMs to have simultaneous, direct access to a single physical GPU, using the same NVIDIA graphics drivers that are deployed on non-virtualized operating systems.
- **Support for VMware vSphere and VMware ESX using Virtual Direct Graphics Acceleration (vDGA)** - You can use HDX 3D Pro with vDGA for both RDS and VDI workloads.

When HDX 3D Pro is used with Virtual Shared Graphics Acceleration (vSGA), support is limited to one monitor. Using vSGA with large 3D models can result in performance issues due to its use of API intercept technology. For more information about issues, refer to [VMware vSphere 5.1 - Citrix Known Issues](https://docs.citrix.com).

HDX 3D Pro integrates with your existing XenDesktop infrastructure. You can deliver graphical applications as part of hosted applications or desktops on Desktop OS machines.

As shown in the following figure:

- The host computer must reside within the same Active Directory domain as your Delivery Controller.
- When a user logs on to Citrix Receiver and accesses the virtual application or desktop, the controller authenticates the user and contacts the VDA for HDX 3D Pro to broker a connection to the computer hosting the graphical application. The VDA for HDX 3D Pro uses the appropriate hardware on the host to compress views of the complete desktop or just of the graphical application.
- The desktop or application views and the user interactions with them are transmitted between the host computer and the user device through a direct HDX connection between Citrix Receiver and the VDA for HDX 3D Pro.
Install and configure HDX 3D Pro for Windows Desktop OS

Dec 03, 2013

Pre-requisites

- An installed and configured XenDesktop infrastructure
- A XenDesktop site
- If using XenDesktop with XenServer vGPU, refer to
  — vGPU Release Notes
  and
  — Configuring XenServer to use NVIDIA GRID
  on Citrix Virtual GPU Solution.

To install using the graphical interface

To enable users to connect to the physical machine or XenServer VM hosting the graphical application, you install the Virtual Delivery Agent (VDA) for HDX 3D Pro. Then, to assign the virtual desktop or application to a user, you create a machine catalog and a delivery group containing the computer that hosts the graphical application.

1. Prepare the VM or physical machine that will host the graphical application: Install and set up the graphical application, plus any other applications that are required.
   For more information about assigning a GPU to a Windows VM, see the XenServer Virtual Machine User's Guide.
   Note: By default, each virtual CPU that you allocate to a XenServer VM is assigned to a single-core socket. This means that on operating systems with socket restrictions, you can only use a limited number of the CPU cores on the host server. XenServer Advanced, Enterprise, and Platinum editions include a feature that enables you to specify the number of cores per virtual CPU in a VM. For more information, see http://support.citrix.com/article/CTX126524.

2. For user connections to physical VDAs with physical monitors attached, verify that the user device resolution is equal to or less than the VDA resolution.
   If the resolution requested by the user device is high, the ICA session will not detect the resolution of the user device and will fall back to the minimum resolution supported by the VDA monitor.

3. Join the host computer to the Active Directory domain containing your Delivery Controller. Make a note of the Active Directory computer account name for the host computer as you will need to know this when you create a machine catalog.

4. Insert the XenDesktop installation media into the optical drive or mount the ISO on the host computer. If autorun is not enabled, navigate to and run AutoSelect.exe on the installation media.

5. In the XenDesktop installation wizard, select Virtual Delivery Agent for Windows Desktop OS or Virtual Delivery Agent for Windows Server OS.

6. On the Environment page, choose to Create a Master Image (if you use Machine Creation Services or Provisioning Services to create virtual desktops) or Enable Remote PC Access (if you want users to connect to an existing machine).

7. On the Core Components page, click Next.

8. On the Delivery Controller page, specify the controllers in the XenDesktop site to which the VDA for HDX 3D Pro will connect, either by manually entering the locations, by selecting controllers from Active Directory, or by allowing Machine Creation Services to specify the Controllers. Alternatively, select Do it later if you plan to specify controller locations later using Group Policy or by running the installer again.
Important: Ensure that you specify the locations of all the controllers in the site, otherwise some user connections may be refused. For load balancing, the VDA for HDX 3D Pro automatically distributes connections evenly across the controllers.

9. If you are using a firewall other than Windows Firewall on the host computer, manually enable ports 80, 1494, 2598, and 3389 to allow XenDesktop to function correctly and open ports 16500–16509 to enable Real-time Transport for Audio. If Windows Firewall is running on the host computer, the installer gives you the option to open the ports automatically. Click Next.

10. On the Summary page, click Install.

Before the VDA for HDX 3D Pro is installed, the following prerequisites are installed if they are not already present on the host computer.

- Microsoft .NET Framework 3.5 SP 1
- Microsoft Visual C++ 2005, 2008 SP 1, and 2010 Redistributable packages

11. When the installation is complete, ensure that the Restart machine (required to complete install) check box is selected and click Close.

12. After completing the installation of the VDA for HDX 3D Pro, log on to the computer running Studio, create a machine catalog, and add the computer hosting the graphical application. For help with this step, see Create a new machine catalog.

When creating a machine catalog for vGPU-enabled desktops, be sure to use the following settings:

- On the Operating System and Hardware page, select Windows Desktop OS.
- On the Machine Management page, select Virtual machines.
- On the Desktop Experience page, select I want users to connect to the same (static) desktop each time they log on.

13. To make the hosted desktops on Desktop OS machines available to users, create a desktop delivery group using the machine catalog containing the host computer. For help with this step, see Create a new Delivery Group.

To install using the command line

You can also install the VDA for HDX 3D Pro from a command prompt. To install HDX 3D Pro, run XenDesktopVdaSetup.exe and include the following option in addition to any others you need to use.

- /ENABLE_HDX_3D_PRO

For information about other options, run XenDesktopVdaSetup.exe /h or see Install using the command line.

To deploy the VDA for HDX 3D Pro through Active Directory Group Policy, ensure that the transform file specifies appropriate values for the ENABLE_HDX_3D_PRO properties. For more information about deploying the VDA through group policy, see Install or remove Virtual Delivery Agents using scripts in Active Directory.

To upgrade HDX 3D Pro

To upgrade HDX 3D Pro, uninstall both the separate HDX 3D for Professional Graphics component and the VDA before installing the latest VDA for HDX 3D Pro. Similarly, to switch from the standard VDA to the one for HDX 3D Pro, uninstall the standard VDA and then install the VDA for HDX 3D Pro.
Install and upgrade NVIDIA drivers

Oct 10, 2013
The NVIDIA GRID API provides direct access to the frame buffer of the GPU, providing the fastest possible frame rate for a smooth and interactive user experience. If you install NVIDIA drivers before you install XenDesktop VDA with HDX 3D Pro, NVIDIA GRID is enabled by default.

To enable NVIDIA GRID on a VM, disable Microsoft Basic Display Adapter from the Device Manager. Run the following command and then restart the VDA:

Montereyenable.exe --enable --noreset

If you install NVIDIA drivers after you install XenDesktop VDA with HDX 3D Pro, NVIDIA GRID is disabled. Enable NVIDIA GRID by using the Montereyenable tool provided by NVIDIA.

To enable NVIDIA GRID, run the following command, then restart the VDA:

Montereyenable.exe --enable --noreset

To disable NVIDIA GRID, run the following command, then restart the VDA:

Montereyenable.exe --disable --noreset
Optimize the HDX 3D Pro user experience

Oct 10, 2013
Implement the following recommendations to provide the optimum experience for your users:

- To use HDX 3D Pro with multiple monitors, ensure that the host computer is configured with at least as many monitors as are attached to user devices. The monitors attached to the host computer can be either physical or virtual.
- Do not attach a monitor (either physical or virtual) to a host computer while a user is connected to the virtual desktop or application providing the graphical application. Doing so can cause instability for the duration of a user's session.
- Let your users know that changes to the desktop resolution (by them or an application) are not supported while a graphical application session is running. After closing the application session, a user can change the resolution of the Desktop Viewer window in the Citrix Receiver – Desktop Viewer Preferences.
- When multiple users share a connection with limited bandwidth, such as at a branch office, Citrix recommends that you use the Overall session bandwidth limit policy to limit the bandwidth available to each user. This ensures that the available bandwidth does not fluctuate widely as users log on and off. Because HDX 3D Pro automatically adjusts to make use of all the available bandwidth, large variations in the available bandwidth over the course of user sessions can negatively impact performance. For more information, see the Bandwidth policy settings. For example, if 20 users share a 60 Mbps connection, the bandwidth available to each user can vary between 3 Mbps and 60 Mbps depending on the number of concurrent users. To optimize the user experience in this scenario, determine the bandwidth required per user at peak periods and limit users to this amount at all times.
- For users of a 3D mouse, Citrix recommends that you increase the priority of the Generic USB Redirection virtual channel to 0. For information about changing the virtual channel priority, see http://support.citrix.com/article/CTX128190.
GPU acceleration for Windows Server OS

Mar 07, 2016

HDX 3D Pro allows graphics-heavy applications running in Windows Server OS sessions to render on the server's graphics processing unit (GPU). By moving OpenGL, DirectX, Direct3D, and Windows Presentation Foundation (WPF) rendering to the server's GPU, the server's central processing unit (CPU) is not slowed by graphics rendering. Additionally, the server is able to process more graphics because the workload is split between the CPU and GPU.

When using HDX 3D Pro, multiple users can share graphics cards. When HDX 3D Pro is used with XenServer GPU Passthrough, a single server hosts multiple graphics cards, one per virtual machine.

GPU Sharing for RDS workloads

GPU Sharing enables GPU hardware rendering of OpenGL and DirectX applications in remote desktop sessions. GPU Sharing has the following characteristics:

- Can be used on bare metal or virtual machines to increase application scalability and performance.
- Enables multiple concurrent sessions to share GPU resources. (Most users do not require the rendering performance of a dedicated GPU.)
- Requires no special settings.

Graphics cards

You can install multiple GPUs on a hypervisor and assign VMs to each of these GPUs on a one-to-one basis:

- Install a graphics card with more than one GPU.
- Or, install multiple graphics cards with one or more GPUs each.

Mixing heterogeneous graphics cards on a server is not recommended.

Note: Virtual machines require direct passthrough access to a GPU, which is available with Citrix XenServer or VMware vSphere. When HDX 3D Pro is used in conjunction with GPU Passthrough, each GPU in the server supports one multi-user virtual machine.

GPU Sharing does not depend on any specific graphics card.

- When running on a hypervisor, select a hardware platform and graphics cards that are compatible with your hypervisor's GPU Passthrough implementation. The list of hardware that has passed certification testing with XenServer GPU Passthrough is available at GPU Passthrough Devices.
- When running on bare metal, it is recommended to have a single display adapter enabled by the operating system. If multiple GPUs are installed on the hardware, disable all but one of them using Device Manager.

Scalability

Scalability using GPU Sharing depends on these factors:

- The applications being run
- The amount of video RAM they consume
- The graphics card's processing power

For example, scalability figures in the range of 8-10 users have been reported on NVIDIA Q6000 and M2070Q cards running applications such as ESRI ArcGIS. These cards offer 6 GB of video RAM. Newer NVIDIA GRID cards offer 8 GB of video RAM and significantly higher processing power (more CUDA cores). With the NVIDIA GRID K2 cards, good performance has been
observed with up to 20 users per GRID K2 card. Other applications may scale much higher, achieving 32 concurrent users on a high-end GPU.

Note: Some applications handle video RAM shortages better than others. If the hardware becomes extremely overloaded, this could cause instability or a crash of the graphics card driver. Limit the number of concurrent users to avoid such issues. To confirm that GPU acceleration is occurring, use a third-party tool such as GPU-Z. GPU-Z is available at http://www.techpowerup.com/gpuz/.

DirectX, Direct3D, and WPF rendering

DirectX, Direct3D, and WPF rendering is only available on servers with a GPU that supports a display driver interface (DDI) version of 9ex, 10, or 11.

- On Windows Server 2008 R2, DirectX and Direct3D require no special settings to use a single GPU.
- On Windows Server 2012, Remote Desktop Services (RDS) sessions on the RD Session Host server use the Microsoft Basic Render Driver as the default adapter. To use the GPU in RDS sessions on Windows Server 2012, enable the Use the hardware default graphics adapter for all Remote Desktop Services sessions setting in the group policy Local Computer Policy > Computer Configuration > Administrative Templates > Windows Components > Remote Desktop Services > Remote Desktop Session Host > Remote Session Environment.
- To enable WPF applications to render using the server's GPU, create the following settings in the registry of the server running Windows Server OS sessions:
  - [HKEY_LOCAL_MACHINE\SOFTWARE\Citrix\CtxHook\AppInit_Dlls\Multiple Monitor Hook]
    "EnableWPFHook"=dword:00000001
  - [HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Citrix\CtxHook\AppInit_Dlls\Multiple Monitor Hook]
    "EnableWPFHook"=dword:00000001

Experimental GPU acceleration for CUDA or OpenCL applications

This release also provides experimental support for GPU acceleration of CUDA and OpenCL applications running in a user session. This support is disabled by default, but you can enable it for testing and evaluation purposes. Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

1. To use the experimental CUDA acceleration features, enable the following Registry settings:
   - [HKEY_LOCAL_MACHINE\SOFTWARE\Citrix\CtxHook\AppInit_Dlls\Graphics Helper] "CUDA"=dword:00000001
   - [HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Citrix\CtxHook\AppInit_Dlls\Graphics Helper]
     "CUDA"=dword:00000001

2. To use the experimental OpenCL acceleration features, enable the following Registry settings:
   - [HKEY_LOCAL_MACHINE\SOFTWARE\Citrix\CtxHook\AppInit_Dlls\Graphics Helper] "OpenCL"=dword:00000001
   - [HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Citrix\CtxHook\AppInit_Dlls\Graphics Helper]
     "OpenCL"=dword:00000001
OpenGL Software Accelerator

Apr 02, 2015

The OpenGL Software Accelerator is a software rasterizer for OpenGL applications such as ArcGIS, Google Earth, Nehe, Maya, Blender, Voxler, CAD, and CAM. In some cases, the OpenGL Software Accelerator can eliminate the need to use graphics cards to deliver a good user experience with OpenGL applications.

Important: The OpenGL Software Accelerator is provided "as is", and must be tested with all applications. It might not work with some applications and is intended as a solution to try if the Windows OpenGL rasterizer does not provide adequate performance. If the OpenGL Software Accelerator works with your applications, it can be used as a way to avoid the cost of GPU hardware.

Supported platforms:

- Windows 8, 64-bit and 32-bit editions
- Windows 7, 64-bit and 32-bit editions
- Windows Server 2012 R2
- Windows Server 2012
- Windows Server 2008 R2

When should you try the OpenGL Software Accelerator?

- If the performance of OpenGL applications running in virtual machines on XenServer or other hypervisors is an issue, try using OpenGL Accelerator. For some applications, the OpenGL Accelerator outperforms the Microsoft OpenGL software rasterizer that is included with Windows because the OpenGL Accelerator leverages SSE4.1 and AVX. OpenGL Accelerator also supports applications using OpenGL versions up to 2.1.
- For applications running on a workstation, first try the default version of OpenGL support provided by the workstation's graphics adapter. If the graphics card is the latest version, in most cases it will deliver the best performance. If the graphics card is an old version or does not deliver satisfactory performance, then try the OpenGL Software Accelerator.
- 3D OpenGL applications that are not adequately delivered using CPU-based software rasterization may benefit from OpenGL GPU hardware acceleration. This feature can be used on bare metal or virtual machines. For more information, see Open GL GPU Sharing.
Optimize audio features

Jun 27, 2013
To optimize HDX audio features, use the following Citrix policy settings:

- Audio quality
- Client audio redirection
- Client microphone redirection
- Audio Plug N Play
- Audio redirection bandwidth limit
- Audio redirection bandwidth limit percent
- Audio over UDP Real-time Transport
- Audio UDP port range

Important: Most audio features are transported using the ICA stream and are secured in the same way as other ICA traffic. User Datagram Protocol (UDP) audio uses a separate, unsecured, transport mechanism.

Select audio quality

In general, higher sound quality consumes more bandwidth and greater server CPU utilization by sending more audio data to user devices. Sound compression allows you to balance sound quality against overall session performance; use Citrix policy settings to configure the compression levels to apply to sound files.

By default, Audio quality is set to High - high definition audio. This setting provides high fidelity stereo audio; however, it consumes more bandwidth than other quality settings.

Note: Do not use this audio quality for non-optimized voice chat or video chat applications (such as softphones), as it may introduce latency into the audio path that is not suitable for real-time communications.

Consider creating separate policies for groups of dial-up users, and for those who connect over a LAN or WAN. Over dial-up connections, where bandwidth typically is limited, users are likely to care more about download speed than sound quality. You may therefore want to create one policy for dial-up connections that applies high compression levels to sound, and another for LAN or WAN connections that applies lower compression levels.

To change audio quality, add the Audio quality setting to a policy and select one of the following audio quality levels:

- Medium - optimized for speech. Use this setting to deliver Voice over IP (VoIP) applications, or to deliver media applications in challenging network connections with very low (less than 512Kbps) lines or significant congestion and packet loss. Audio sent to the user device is compressed up to 64Kbps; this compression results in a moderate decrease in the quality of the audio played on the user device, while providing low latency and consuming very low bandwidth. If VoIP quality is unsatisfactory with Audio quality set to Medium, ensure that Audio over UDP Real-time Transport policy setting is set to Allowed.
  Note: UDP/RTP requires Audio quality to be Medium.
- Low - for low-speed connections. Sounds sent to the user device are compressed up to 16Kbps; this compression results in a significant decrease in the quality of the sound, but provides reasonable performance for a low-bandwidth connection.

Important: Remember to enable audio on Client audio settings on the user device, as described later in this topic.

Redirect audio reception

You can allow users to receive audio from an application on a server through speakers or other sound devices (such as headphones) on the user device. By default, Client audio redirection is Allowed.
Important: To use this feature, remember to enable audio on Client audio settings on the user device, as described later in this topic. On Windows Server OS machines, also ensure that Audio Plug N Play is Enabled to support multiple audio devices. Client audio mapping may cause a heavy load on the servers and the network; to turn off this feature, add the Client audio redirection setting to a policy and set its value to Prohibited.

Important: When Client audio redirection is Prohibited, all HDX audio functionality is disabled. Activate user device microphones

You can allow users to record audio using input devices such as microphones on the user device. By default, Client microphone redirection is Allowed; however, if audio is disabled on the client software, or if Client audio redirection is Prohibited, this setting has no effect.

To record audio, the user device needs either a built-in microphone or a device that can be plugged into the microphone jack or USB port. For security, users are alerted when servers that are not trusted by their user devices try to access microphones, and can choose to accept or reject access prior to using the microphone. Users can disable this alert on Citrix Receiver.

Important: To use this feature, remember to enable audio on Client audio settings on the user device, as described later in this topic. On Windows Server OS machines, also ensure that Audio Plug N Play is Enabled to support multiple audio devices. To turn off this feature, add the Client microphone redirection setting to a policy and set its value to Prohibited.

Allow multiple audio devices

The Audio Plug N Play policy setting controls whether to allow the use of multiple audio devices to record and play sound. This setting is Enabled by default; to disable this feature, add the Audio Plug N Play setting to a policy and set its value to Disabled.

Note: This setting applies only to Windows Server OS machines.

Set audio redirection bandwidth limits

The Audio redirection bandwidth limit policy setting specifies the maximum bandwidth (in kilobits per second) for a playing and recording audio in a session. The Audio redirection bandwidth limit percent setting specifies the maximum bandwidth for audio redirection as a percentage of the total available bandwidth. By default, zero (no maximum) is specified for both settings.

To configure the Audio redirection bandwidth limit or Audio redirection bandwidth limit percent or both, add the setting to a policy and type a number in the corresponding Value field. If both settings are configured, the one with the lowest bandwidth limit is used.

Important: To use this feature, remember to enable audio on Client audio settings on the user device, as described later in this topic.

Send and receive audio with UDP

By default, Audio over UDP Real-time Transport is Allowed, opening up a UDP port on the server to support all connections configured to use Audio over UDP Real-Time Transport. Citrix recommends configuring UDP/RTP for audio, to ensure the best possible user experience in the event of network congestion or packet loss.

Important: Audio data transmitted with UDP is not encrypted.

By default, UDP audio on XenDesktop uses two consecutive ports within the range of ports 16500 to 16509 to pass through the Windows firewall; the default ports are 16500, 16501. To use other ports, add the Audio UDP port range setting to a policy and type the port number or range (in the form lowest port number: highest port number) into the Value field. This setting specifies the range of port numbers used by the Virtual Delivery Agent (VDA) to exchange audio packet data...
with the user device. The VDA attempts to use each UDP port pair to exchange data with the user device, starting with the lowest and incrementing by 2 for each subsequent attempt. Each specified port handles both inbound and outbound traffic.

Important: To use audio over UDP, remember to enable audio on Client audio settings on the user device, as described below.

Configure audio setting policies for user devices

2. For Client audio settings, select Not Configured, Enabled, or Disabled.
3. If you selected Enabled, choose a sound quality. For UDP audio, use Medium (the default) only.
4. For UDP audio only, select Enable Real-Time Transport and then set the range of incoming ports to use to pass through the local Windows firewall.

Avoid echo during multimedia conferences

When users take part in audio or video conferences, they may hear an echo in their audio. Echoes usually occur when speakers and microphones are too close to each other. For that reason, Citrix recommends the use of headsets for audio and video conferences.

HDX provides an echo cancellation option, enabled by default, which minimizes echo during a conference. The effectiveness of echo cancellation is sensitive to the distance between the speakers and the microphone. These devices must not be too close to each other, or too far away from each other.

To disable echo cancellation:

1. Navigate to the applicable registry entry.

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

- **For 32-bit computers:** On the user device, open the registry and navigate to
  HKEY_LOCAL_MACHINE\SOFTWARE\Citrix\ICA\Client\Engine\Configuration\Advanced\Modules\ClientAudio\EchoCancellation.
- **For 64-bit computers:** On the user device, open the registry and navigate to
  HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Citrix\ICA\Client\Engine\Configuration\Advanced\Modules\ClientAudio\EchoCancellation.
2. Change the Value data field to FALSE.
Assign priorities to network traffic

Apr 03, 2013

Priorities are assigned to network traffic across multiple connections for a session with quality of service (QoS)-supported routers. Four TCP/IP streams (real-time, interactive, background, and bulk) and one UDP/RTP stream (for voice) are available to carry ICA traffic between the user device and the server. Each virtual channel is associated with a specific priority and transported in the corresponding connection. You can set the channels independently, based on the TCP port number used for the connection. The four priorities are:

- **Very High** – For real-time activities, such as webcam conferences.
- **High** – For interactive elements, such as the screen, keyboard, and mouse.
- **Medium** – For bulk processes, such as client drive mapping.
- **Low** – For background activities, such as printing.

XenDesktop supports multiple channel streaming connections for Virtual Delivery Agents (VDAs) installed on Windows 8 and Windows 7 environments. Work with your company’s network administrator to ensure the Common Gateway Protocol (CGP) ports configured in the Multi-Port Policy setting are assigned correctly on the network routers.

The Secure Sockets Layer (SSL) connections are only supported when the connections are traversing a NetScaler Gateway that supports multi-stream. When running on an internal corporate network, multi-stream connections with SSL are not supported.

Quality of service (QoS) is supported only when multiple session reliability ports, or the CGP ports, are configured.

Caution: Use transport security when using this feature. Citrix recommends using Internet Protocol Security (IPsec) or Secure Sockets Layer (SSL).

To set quality of service for multiple streaming connections

Add the following settings to a policy.

1. **Multi-Port Policy** – This setting specifies ports for ICA traffic across multiple connections, and establishes network priorities.
   1. Select a priority from the CGP default port priority list; by default, the primary port (2598) has a High priority.
   2. Type additional CGP ports in CGP port1, CGP port2, and CGP port3 as needed, and identify priorities for each; each port must have a unique priority.
      
      Important: Firewalls on VDAs must be explicitly configured to allow the additional TCP traffic.

2. **Multi-Stream** – This setting is Disabled by default because it is not needed when using a branch repeater that supports Multi-Stream. Select Enabled:
   - To achieve the desired QoS when using legacy branch repeaters or third-party routers.
   - To enable the Multi-Stream feature for specific users.

Important: For the policies to take effect, users must log off and then log on to the network.
Configure file access for mapped client drives

Apr 09, 2013
You control whether users can copy files from their virtual environments to their user devices. By default, files and folders on mapped client-drives are available in read/write mode from within the session.

To prevent users from adding or modifying files and folders on mapped client-devices, enable the Read-only client drive access policy setting.
Important: When adding this setting to a policy, make sure the Client drive redirection setting is also present in the policy and is set to Allowed.
Enhance the user experience for mobile devices

Mar 04, 2013

XenDesktop delivers a superior user experience for mobile users, often with no special configuration required. Integrate XenDesktop with other Citrix products to provide additional features to your mobile workforce.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Provided by:</th>
</tr>
</thead>
<tbody>
<tr>
<td>High definition user experience on 3G and 4G networks</td>
<td>HDX in XenDesktop&lt;br&gt;In most cases, the default Citrix policy settings provide the best user experience.</td>
</tr>
<tr>
<td>Touch-friendly interface to virtual desktops and applications, optimized for tablet devices</td>
<td>Mobile Experience policy settings in XenDesktop&lt;br&gt;The default settings generally provide the best user experience.</td>
</tr>
<tr>
<td>Remote access to office PCs from other devices, including smart phones, tablets, laptops, and PCs</td>
<td>Remote PC Access in XenDesktop&lt;br&gt;Citrix Receiver</td>
</tr>
<tr>
<td>Automatically provisioned applications to all users of a store</td>
<td>Citrix StoreFront&lt;br&gt;Citrix App Controller</td>
</tr>
<tr>
<td>Email-based account discovery, enabling users to set up an account by entering their email address</td>
<td>Citrix StoreFront&lt;br&gt;Citrix NetScaler Gateway (for remote connections)</td>
</tr>
<tr>
<td>Secure access to stores, desktops and applications</td>
<td>Citrix StoreFront&lt;br&gt;Citrix NetScaler Gateway</td>
</tr>
<tr>
<td>Simplified print management that allows network printing from any device</td>
<td>Universal Print Server in XenDesktop</td>
</tr>
<tr>
<td>Optimized performance and delivery of services to branch offices and mobile users</td>
<td>Citrix CloudBridge</td>
</tr>
<tr>
<td>Role-based management, configuration, and security for corporate and employee-owned mobile devices</td>
<td>Mobile Solutions Bundle (XenMobile MDM and CloudGateway)</td>
</tr>
</tbody>
</table>
Mobility features

May 21, 2014

The Mobile SDK for Windows Apps provides a mobile device programming interface for Windows programs hosted on Citrix environments and delivered to any device with Citrix Receiver. The SDK provides Enterprise Windows developers with the means to develop Windows applications with capabilities and behaviors typical of native mobile applications. The functionality provided by the SDK significantly improves the user experience.

Developers can use the SDK with the Windows application development framework that best suits their needs. Download the SDK from the Citrix download pages.

Typical Windows applications are based on an expansive desktop space with access to a keyboard and a mouse. Legacy Windows applications do not accommodate on-screen keyboards and other features specific to mobile devices. The Mobile SDK for Windows Apps enables Windows developers to write new applications and to improve existing applications for delivery to supported mobile devices.

The Mobile SDK for Windows Apps enables a Windows application to control mobile device features such as:

- **Buttons** – Get the current button target, set the button target, and specify whether to handle a button press on the server or the mobile device.
- **Camera** – Take, download, and remove pictures using the built-in camera of the mobile device. Get picture state information.
- **Device properties** – Retrieve device feature support information.
- **Display** – Get and set mobile device screen metrics such as orientation, scroll mode, and viewport origin to ensure that text is easy to read and controls are easy to use.
- **Keyboard** – Check the keyboard state and control whether to show or hide the on-screen keyboard. The keyboard state includes the keyboard type, keyboard flags, auto-capitalization, return key type, and edit field rectangle.
- **Notification** – Notify the user about special events using sound, vibration, light, and text.
- **Phone** – Make phone calls based on the contacts list on a server.
- **Picker control** – Select an item from a list using a control that is native to the device.
- **SMS** – Send an SMS from content on a server.

For features such as phone calls, SMS, and camera functions enabled by the Mobile SDK for Windows Apps, Receiver prompts the user for permission to perform the action so that the user always has the option to protect potentially sensitive information.

When developing hosted applications that use the Mobile SDK for Windows Apps, consider the following:

- A secure connection (for example, using SSL/TLS or a VPN) should be enforced for the applications. Citrix Receiver should connect to trusted servers.
- Consider obtaining legal advice regarding the use of mobile device features, such as the camera, which raise privacy issues.

Learn more about the Mobile SDK for Windows Apps [here](https://docs.citrix.com).
About mobility features

Jun 20, 2013

XenDesktop mobility features improve the experience of your mobile device users accessing your published resources. Features include:

- The use of mobile device controls instead of native Windows controls such as combo boxes.
- Automatic display of the device keyboard when an editable field has the focus. The desktop session scrolls if needed to make the input area visible.
- A touch-optimized desktop for mobile devices that provides:
  - Improved access to the Windows Start menu: Tap the START button and use the touch-friendly menus to navigate to applications and documents. Start an application or open a document with a single tap.
  - Multiple pages of icons on the desktop: Swipe the desktop or tap the scroll icons to navigate.
  - One-tap return to the touch-optimized desktop when it is hidden by a full-screen application: Tap the icon in the bottom left corner of the desktop.
  - One-tap return to the traditional Windows desktop: Tap the icon in the top right corner of the desktop to toggle between the touch-optimized and Windows desktops.
- Support for providing mobile device location (GPS) information to remote application sessions. This feature enables the remote application to obtain mobile device location information from Citrix Receiver so that the application behavior can change just as if it were running locally on the mobile device.
- A mobile device development platform, the Mobile SDK for Windows Apps, that enables Enterprise Windows developers to write applications for mobile devices using familiar programming languages. The Mobile SDK for Windows Apps includes interfaces to:
  - Control how buttons are used on the mobile device
  - Set screen orientation
  - Activate the on-screen keyboard
  - Use local user interface controls instead of Windows controls
  - Access the device's telephone, SMS, and camera functions

Please note that this release of XenDesktop does not support the audio and video capture capabilities in the latest release of the Mobile SDK for Windows Apps.

Fixed Issues

- The automatic keyboard and mobile combo box now appear on second use. [253264]
- The policy help text for Automatic keyboard display and Remote the combo box is correct. [256356]
- The error "wfshell shell has stopped working" no longer appears when you start Microsoft Notepad after previously exiting it before the keyboard opened. [261973]

Known Issues

- An application with a .NET 4.0 Calendar control can crash if Microsoft UI Automation monitoring is active in the same session and the user places the focus on the Calendar control. This issue results from a missing property (ComponentResourceKey) in the DataTemplate key for the .NET 4.0 Calendar control. A resource defined at the theme level must use a ComponentResourceKey as the key.
  To avoid this issue with .NET 4.0, set the DataTemplate key for the Calendar control as follows:

```xml
<DataTemplate x:Key="{ComponentResourceKey
```
The automatic keyboard feature does not scroll the display to show the input area when an iOS device resolution is set to a value other than Auto-Fit. [267307]

During some operations, applications can unexpectedly minimize to the touch-optimized desktop taskbar. Tap the taskbar icon for the application to re-open it. [267606, 267609]

When a Windows notification appears, the Windows taskbar displays on top of the touch-optimized desktop taskbar. To redisplay the touch-optimized desktop taskbar, dismiss the notification or tap the desktop. [268911]

The touch-optimized desktop taskbar appears with the Windows desktop when the user presses Alt-Tab and then taps the gear icon from the Windows desktop (Receiver for iOS). To correct the display, tap the icon in the lower left corner. [269535]

The keyboard covers the input area when the automatic keyboard is displayed and a Receiver for iOS user rotates the device. The user can pan the display or rotate the device to the original orientation to see the input area. [269920]

The touch-optimized desktop taskbar displays when an application is running in full-screen mode. [272692]

The automatic keyboard or device-native combo box do not display for applications run with elevated permissions (Receiver for iOS). [273016]
System requirements for mobility features

Jun 20, 2013

Environments

- Citrix XenDesktop 7 for Windows Server 2008 R2 SP1 with Microsoft .NET Framework 3.5 SP1
- Citrix XenDesktop 7 for Windows Server 2012 with Microsoft .NET Framework 4.0
- Citrix XenDesktop 7 for Windows 7 and Windows 8 with Microsoft .NET Framework 4.0

Devices

- Citrix Receiver for Android 3.x
- Citrix Receiver for iOS 5.5.x, 5.6.x, 5.7, and 5.8
- Citrix Receiver for Windows 8/RT 1.2 and 1.3

Application

- Location sensing is supported for applications that use the Windows 7 Location API and can receive responses based on the client location sensor.

Mobile SDK for Windows Apps

- Development operating system: Microsoft Windows 7 (x64) or Windows 8
- Development platforms: Microsoft Visual Studio 2010 SP1 or 2012
- Microsoft .NET Framework 3.5 SP1 and 4.0
- Microsoft Windows SDK 7.1 (for C++ location support)
Configure policies for mobility features

Jun 20, 2013

The following Citrix user configuration policy settings control mobility feature settings.

Under ICA > Mobile Experience:
• Automatic keyboard display
• Launch touch-optimized desktop
• Remote the combo box

Under ICA > Client Sensors > Location:
• Allow applications to use the physical location of the client device

To set the keyboard display behavior

The **Automatic keyboard display** policy setting determines the behavior of the keyboard during application sessions on mobile devices. By default, a mobile Receiver user must manually open the keyboard. To enable the keyboard to automatically open when an editable field has the focus, set this policy to Allowed. When this setting is allowed, a user can change a Receiver for iOS session setting to prevent the keyboard from automatically opening.

To provide a touch-friendly interface

The **Launch touch-optimized desktop** policy setting determines the overall Receiver interface behavior. By default, a touch-friendly interface that is optimized for tablet devices is used. To use only the Windows interface, set this policy to Prohibited.

To set the type of combo box displayed

The **Remote the combo box** policy setting determines the type of combo box displayed during application sessions on mobile devices. To display the device-native combo box control, set this policy to Allowed. When this setting is allowed, a user can change a Receiver for iOS session setting to use the Windows combo box.

To allow applications to use mobile device location information

The **Allow applications to use the physical location of the client device** policy setting determines whether applications running in a XenApp session on a mobile device are allowed to use the physical location of the client device.

By default, the use of location information is prohibited. To allow use of location information, set this policy to Allowed. When this setting is allowed, a user can prohibit use of location information by denying a Receiver request to access the location. Android and iOS devices prompt at the first request for location information in each session.
Provide users with Remote PC Access

Jun 17, 2013
Refer to Remote PC Access for an introduction to this feature and a list of considerations to review when planning your deployment.

To provide users with remote access to their office PCs:

1. Use Citrix Studio to initially configure your XenDesktop site as a remote PC deployment. When you create a site, a wizard guides you through the process of creating the first machine catalog and Delivery Group and mapping users to those machines. Enter the following information:

<table>
<thead>
<tr>
<th>On this page:</th>
<th>You must:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System and Hardware</td>
<td>Select Remote PC Access.</td>
</tr>
<tr>
<td>Machine Accounts</td>
<td>Select from existing machine accounts or Organizational Units (OUs) that appear on the page. You can provide additional for Remote PC Access machine accounts using Add machine accounts or Add OUs. Selecting one of these options displays a dialog box that lets you select Domain folders and subfolders OUs.</td>
</tr>
<tr>
<td>Scopes</td>
<td>Select or change the scope.</td>
</tr>
<tr>
<td></td>
<td>Scopes are groups of objects that represent parts of your deployment and your organization. For example, you might have a scope for Delivery Groups used by a Sales team, and another that you assign to the Support team. Make sure that you select the correct scopes so that the right users can access the desktops in this Delivery Group.</td>
</tr>
<tr>
<td>Summary</td>
<td>Review information you entered and add a description for users and a description for administrators.</td>
</tr>
</tbody>
</table>

Note: Choosing a Remote PC deployment will not prevent VDI use of the site in the future.

2. Install the VDA on the office PC used for local and remote access. Typically, you deploy the VDA automatically using your package management software; however, for proof-of-concept or small deployments, you can install the VDA manually on each office PC.

Once the VDA is installed, the next domain user that logs on to a console session (locally or via RDP) on the office PC is automatically assigned to the Remote PC desktop. If additional domain users log on to a console session, they are also added to the desktop user list, subject to any restrictions you have configured.
Note: To use RDP connections outside of your XenDesktop environment, you must add users or groups to the Direct Access Users group.

3. Instruct your users to download and install Citrix Receiver onto each client device that accesses the office PC remotely. Citrix Receiver is available from http://www.citrix.com or the application distribution systems for supported mobile devices.

Desktop users can securely access resources on the office PC while experiencing the benefits of Citrix HDX technology.

Troubleshooting Remote PC Access

The Delivery Controller writes the following diagnostic information about Remote PC Access to the Windows Application Event log:

<table>
<thead>
<tr>
<th>Event ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Informational:</strong></td>
</tr>
<tr>
<td>3300</td>
<td>Machine added to catalog</td>
</tr>
<tr>
<td>3301</td>
<td>Machine added to delivery group</td>
</tr>
<tr>
<td>3302</td>
<td>Machine assigned to user</td>
</tr>
<tr>
<td></td>
<td><strong>Error:</strong></td>
</tr>
<tr>
<td>3303</td>
<td>Exception</td>
</tr>
</tbody>
</table>

Note: Informational messages are not throttled. Error messages are throttled by discarding duplicate messages.
Prevent user access to the local desktop

Mar 21, 2015

Use the Desktop Lock when users do not need to interact with the local desktop. In this access scenario, the virtual desktop effectively replaces the local one, allowing the user to interact with the virtual desktop as if it is local.

The Desktop Lock is a separate component that is released with Citrix XenDesktop and Citrix VDI-in-a-Box. It is an alternative to the Desktop Viewer and is designed mainly for repurposed Windows computers and Windows thin clients. The Desktop Lock employs a replacement Windows shell, in which no Start menu is displayed, to prevent users from accessing the local operating system. A replacement for Windows Task Manager is also used.

With the Desktop Lock, users can access desktops from Server OS and Desktop OS Delivery Groups.

Operating system

The Desktop Lock is supported on the following user device operating systems. Editions or service packs are listed only where support is limited:

- Windows 7, 32-bit and 64-bit editions (including Embedded Edition)
- Windows XP Professional, 32-bit and 64-bit editions
- Windows XP Embedded
- Windows Vista, 32-bit and 64-bit editions

Note: Support for Windows XP ends April 8, 2014 when Microsoft ends extended support for Windows XP. Support for Windows XP Embedded will continue.

Receiver

Citrix Receiver for Windows Enterprise 3.4 package

Environment

User devices running the Desktop Lock are supported only if they are connected to a local area network (LAN).

Domain-joined and Non-domain-joined installations

You can install the Desktop Lock on domain-joined or non-domain-joined user devices. The system behavior and user experience on the different device types are slightly different.

In domain-joined installations, the Windows user device is joined to an Active Directory domain. In these installations:

- User devices access StoreFront stores through XenApp Services URLs. VDI-in-a-Box can also be used.
- Program Neighborhood Agent replaces the Windows Explorer shell, and launches the first alphabetically listed, available desktop in an assigned Desktop Group.
- The Desktop Lock replacement shell does not override the administrator's shell.
- Windows Task Manager is replaced with a version that lets the user restart their virtual desktop or that passes the Ctrl+Alt+Delete keyboard combination to the desktop session.
- On Windows XP user devices, the Ctrl+Alt+Delete keyboard combination summons the replacement Task Manager.
- The user device mimics the reason for a disconnected or terminated session. For example, if the user shuts down the virtual desktop (causing a session to end), the device also shuts down.

Non-domain-joined installations are primarily used by vendors of Windows terminals that are not joined to an Active...
Directory domain. In these installations:

- User devices access StoreFront stores through Desktop Appliance sites.
- User devices are configured to log on the user automatically, and to launch Internet Explorer in Kiosk Mode. The domain logon page is not displayed.
- The only access point is the Desktop Appliance site, which launches the first alphabetically listed, available desktop in an assigned Delivery Group.
- A version of Windows Task Manager passes the Ctrl+Alt+Delete keyboard combination to the desktop session.
Install or remove the Desktop Lock

Mar 12, 2015

User account considerations

When you install the Desktop Lock, the system uses a replacement shell. To allow administration of the user device after you complete the installation, the account used to install CitrixDesktopLock.msi is excluded from the shell replacement. If the account used to install CitrixDesktopLock.msi is later deleted, you will not be able to log on and administer the device.

Citrix does not recommend the use of custom shells because they might interfere with the replacement shell that is used for Desktop Lock sessions.

Users must use domain accounts to access desktops with the Desktop Lock. They cannot use local accounts.

To install the Desktop Lock

Important: Log on using a local administrator account to carry out this installation procedure.

This procedure installs the Receiver for Windows plug-in so that virtual desktops are displayed using the Desktop Lock. Do not use this procedure if you want the Desktop Viewer, which is included in Receiver, to be available to users.

1. At a command prompt, run CitrixReceiverEnterprise.exe using the following syntax. This file is located in the folder called Citrix Receiver and Plug-ins\Windows\Receiver on the installation media:
   
   CitrixReceiverEnterprise.exe ADDLOCAL="ReceiverInside,ICA_Client,SSON,USB,DesktopViewer,Flash,PNAgent,Vd3d" SERVER_LOCATION="my.server" ENABLE_SSON="Yes"

   For information about the properties used in this command, see the topic — Configure and install Receiver for Windows using command-line parameters in the Receiver for Windows documentation.

2. Enter the URL of the XenDesktop Services site where your virtual desktops are located. The URL must be in the format http://servername or https://servername. If you are using hardware or software for load balancing or failover, you can enter a load-balanced address.

   Important: Ensure that the URL you enter is correct. If the URL is incorrectly typed, or you leave the field empty and the user does not enter a valid URL when prompted after installation, no virtual desktop or local desktop will be available.

3. On the XenDesktop installation media, navigate to the Citrix Receiver and Plug-ins\Windows\Receiver folder and double-click CitrixDesktopLock.msi. The Citrix Desktop Lock wizard appears.

4. Read and accept the Citrix license agreement and click Install. The Installation Progress page appears.

5. In the Installation Completed dialog box, click Close.

6. When prompted, restart the user device. If you have been granted access to a desktop and you log on as a domain user, the restarted device is displayed using the Desktop Lock.

To remove the Desktop Lock

If you installed the Desktop Lock, two items are displayed in Add/Remove Programs. You must remove both to complete the removal process.

1. Log on with the same local administrator credentials that were used to install the Desktop Lock.
2. Run the Add/Remove programs utility from the Control Panel.
3. Remove Citrix Desktop Lock.
4. Remove Citrix Receiver or Citrix Receiver (Enterprise).
Configure the Desktop Lock

Aug 22, 2013

Use an administrator account to configure the Desktop Lock. This must be the same account that you used for the installation.

Two .adm files are provided that allow you to do this using policies:

- icaclient.adm. For information on obtaining this file, see To enable smart card usage.
- icaclient_usb.adm. The file is located in the following installed folder: <root drive>:\Program Files\Citrix\ICA Client\Configuration\en.

This topic assumes you have loaded both files into Group Policy, where the policies appear in Computer Configuration or User Configuration > Administrative Templates > Classic Administrative Templates (ADM) > Citrix Components.

To configure USB preferences

When a user plugs in a USB device, that device is automatically remoted to the virtual desktop. No user interaction is required. The virtual desktop is responsible for controlling the USB device and displaying it in the user interface.

1. Turn on USB support in XenDesktop deployments by enabling the USB policy rule. For information on this, see the XenDesktop documentation.
2. In Citrix Receiver > Remoting client devices > Generic USB Remoting, enable and configure as desired the Existing USB Devices, New USB Devices, and USB Devices List In Desktop Viewer policies. You can use the Show All Devices policy to display all connected USB devices, including those using the Generic USB virtual channel (for example, webcams and memory sticks).

To configure drive mapping

In Citrix Receiver > Remoting client devices, enable and configure as desired the Client drive mapping policy.

To configure a microphone

In Citrix Receiver > Remoting client devices, enable and configure as desired the Client microphone policy.

Using the Desktop Lock on multiple monitors

Multiple monitors are fully supported by the Desktop Lock. Up to eight monitors are supported.

A virtual desktop is displayed across all monitors. The primary monitor on the device becomes the primary monitor in the virtual desktop session.

Considerations when configuring the Desktop Lock

Grant access to only one virtual desktop running the Desktop Lock per user.
Do not allow users to hibernate virtual desktops. Use Active Directory policies to prevent this.
Configure smart cards

Mar 28, 2015
This provides an overview of how to prepare Citrix StoreFront, Citrix Desktop Lock, and Desktop Appliances to work with smart cards.

These instruction apply to the version of Desktop Lock included with Citrix Receiver for Windows Enterprise 3.4. To configure the version of Desktop Lock included with Citrix 4.2, see the instructions in Receiver Desktop Lock.

1. Configure StoreFront. See the StoreFront product documentation for details.
   1. Configure the XML Service to use DNS Address Resolution for Kerberos support.
   2. Configure StoreFront sites for HTTPS access, create a server certificate signed by your domain certificate authority, and add HTTPS binding to the default website.

2. Ensure Pass-through with smart card is enabled. This is enabled by default.

3. Enable Kerberos.

4. Enable Kerberos and Pass-through with smart card.

5. Enable Anonymous access on the IIS Default Web Site and use Integrated Windows Authentication.

6. Ensure the IIS Default Web Site does not require SSL and ignores client certificates.

7. Enable XenApp Services support.

2. Configure Local Computer Policies on the user device.
   1. Import the icaclient.adm template using the Group Policy Management Console. The template is available in %Program Files%\Citrix\ICA Client\Configuration\.
   3. Enable Smart card authentication.
   4. Enable Local user name and password.

3. Configure the user device before installing Desktop Lock.
   1. Add the URL for the Delivery Controller to the Windows Internet Explorer Trusted Sites list.
   2. Add the URL for the first desktop group to the Internet Explorer Trusted Sites list. Use the following format: desktop://desktop-20group-20name.
   3. Enable Internet Explorer to use automatic login for Trusted Sites.

4. Configure the user device after installing Desktop Lock.
   1. Edit the registry key, HKEY_LOCAL_MACHINE\SOFTWARE\Citrix\PNAgent\ServerURL to point to the PNAgent config.xml of the Delivery Controller.

   Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

If Citrix Desktop Lock is installed on the user device, a consistent smart card removal policy is enforced. For example, if the Windows smart card removal policy is set to Force logoff for the desktop, the user must log off from the user device as well, regardless of the Windows smart card removal policy set on it. This ensures that the user device is not left in an inconsistent state. This behavior applies only to user devices with the Desktop Lock, not the Desktop Viewer.
Keyboard input in Desktop Lock sessions

Feb 01, 2013

In Desktop Lock sessions, the Windows logo key+L key combination is directed to the local computer. In most cases, Ctrl+Alt+Delete is directed to the local computer. Shift+F2 cannot be used to switch a full-screen desktop session to windowed mode. However, this key combination can be used in any application sessions within the desktop session.

Ctrl+Esc and Alt+Tab are sent to the remote, virtual desktop.

Key presses that activate StickyKeys, FilterKeys, and ToggleKeys (Microsoft accessibility features) are normally directed to the local computer.

Note: The Desktop Lock treats some Windows key combinations differently from the Desktop Viewer. Information on the Desktop Viewer is included in the Receiver documentation.
Server VDI

Aug 14, 2014

Using the Server VDI (Virtual Desktop Infrastructure) feature allows you to deliver a desktop from a server operating system for a single user.

- Enterprise administrators can deliver server operating systems as VDI desktops, which can be valuable for users such as engineers and designers.
- Service Providers can offer desktops from the cloud; those desktops comply with the Microsoft Services Provider License Agreement (SPLA).

You can use the Enhanced Desktop Experience Citrix policy setting to make the server operating system look like a Windows 7 operating system.

The following features cannot be used with Server VDI:

- Personal vDisks
- HDX 3D Pro
- Hosted applications
- Local App Access
- Direct (non-brokered) desktop connections
- Remote PC Access

Server VDI is supported on the same server operating systems as the VDA for Windows Server OS; these are listed in the System requirements topic.

Prepare the server for installation

On the Windows server, ensure that Remote Desktop Services role services are not installed and that users are restricted to a single session:

- Use Windows Server Manager to ensure that the Remote Desktop Services role services are not installed. If they were previously installed, remove them.
- Ensure that the ‘Restrict each user to a single session’ property is enabled.
- On Windows Server 2008 R2, access this property through Administrative Tools > Remote Desktop Services > Remote Desktop Session Host Configuration. In the Edit settings > General section, the Restrict each user to a single session setting should indicate Yes.
- On Windows Server 2012, edit the registry to set the Terminal Server setting. In registry key HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\TerminalServer to set DWORD fSingleSessionPerUser to 1.

Install Microsoft .NET Framework 3.5 SP1 on the server before installing the VDA. (The requirement to install this prerequisite manually applies only to XenDesktop versions 7.0 through 7.5, and XenApp 7.5.)

Install the VDA, then create a Machine Catalog and a Delivery Group

Install a VDA for Windows Desktop OS on a supported server operating system using the installer's command line interface. By default, the installer blocks the Windows Desktop OS VDA on a server operating system using the command line overrides this behavior.

After installing the VDA, you create a new dedicated Machine Catalog for Server VDI. You then assign the new Machine Catalog to a new dedicated Delivery Group.
1. Use the command-line interface to install a VDA on a supported server or server master image, specifying the `/quiet` and `/servervdi` options:
   `XenDesktopVdaSetup.exe /quiet /servervdi`

   Use the `/enable_hdx_ports` option to open ports in the firewall, unless the firewall is to be configured manually.

   Do not include options for features that are not supported with Server VDI, such as `/baseimage`, `/enable_hdx_3d_pro`, or `/xa_server_location`.

   For command information, see Install using the command line.

2. When you create a Machine Catalog for the server where you installed the VDA, specify the following:
   - On the Operating System and Hardware page, select Windows Desktop OS.
   - On the Machine page, add and then select the server or server master image where you installed the Desktop OS VDA.
   - On the Summary page, specify a machine catalog name and description for administrators that clearly identifies it as Server VDI; this will be the only indicator in Studio that the catalog supports Server VDI.

   When using Search in Studio, the Server VDI catalog you created will be displayed on the Desktop OS Machines tab, even though the VDA was installed on a server.

3. Create a Delivery Group and assign the Server VDI catalog you created in the previous step.
Integrate

Mar 12, 2015
This section contains system requirements and deployment information for integrating XenDesktop with the following products:

- **Microsoft System Center Virtual Machine Manager**
  System requirements, deployment, hypervisor configuration and installation, master VM and virtual machine creation

- **Microsoft System Center Configuration Manager**
  XenDesktop properties for Microsoft System Center Configuration Manager

- **VMWare**
  System requirements, hypervisor configuration and installation, and master VM and virtual machine creation with VMWare products
Microsoft System Center Virtual Machine Manager
and virtual machine management

Oct 18, 2013
Configure your system as described in this topic if you use Hyper-V with Microsoft System Center Virtual Machine Manager (VMM) to provide virtual machines in your environment.

System requirements
Before you create virtual machines (VMs), make sure that your environment meets the requirements listed in the Hosts section of System Requirements.

Upgrade from VMM 2012 to VMM 2012 SP1 or VMM 2012 R2
Consider the following component operating system versions combinations when upgrading:

- For VMM and Hyper-V Hosts requirements, see http://technet.microsoft.com/en-us/library/gg610649.aspx.

Note: A mixed Hyper-V cluster is not supported. An example of a mixed cluster is one in which half the cluster is running Hyper-V 2008 and the other is running Hyper-V 2012.

Upgrade from VMM 2008 R2 to VMM 2012 SP1
If you are starting from XenDesktop 5.6 on VMM 2008 R2, it is important to follow this sequence so that XenDesktop can continue to operate without any downtime.

The recommended component upgrade sequence is:

1. Upgrade VMM to 2012 (now running XenDesktop 5.6 and VMM 2012)
2. Upgrade XenDesktop to 7 (now running XenDesktop 7 and VMM 2012)
3. Upgrade VMM from 2012 to 2012 SP1 (now running XenDesktop 7.0 and VMM 2012 SP1)

Upgrade from VMM 2012 SP1 to VMM 2012 R2
If you are starting from XenDesktop 7 on VMM 2012 SP1, it is important to follow this sequence so that XenDesktop can continue to operate without any downtime.

The recommended component upgrade sequence is:

1. Upgrade XenDesktop to 7.1 (now running XenDesktop 7.1 and VMM 2012 SP1)
2. Upgrade VMM 2012 SP1 to 2012 R2 (now running XenDesktop 7.1 and VMM 2012 R2)

Virtual Machine support
Note the following:

- This release supports only Generation 1 virtual machines with VMM 2012 R2. Generation 2 virtual machines are not supported.
- Generation 2 VMs are not supported for both Machine Creation Services (MCS) and Provisioning Services deployments.
When creating VMs in Studio with Machine Creation Services or Provisioning Services, Generation 2 VMs do not appear in the selection list for a master VM.

- Generation 2 VMs have Secure Boot enabled by default, which prevent the VDA for functioning properly.

Install and configure a hypervisor

1. Install Microsoft Hyper-V Server and VMM on your servers. All Delivery Controllers in your environment must be in the same forest as the VMM servers.
2. Install the System Center Virtual Machine Manager Console on all Delivery Controllers in your machine environment.
3. Verify the following account information:
   - The account you use to create hosts in Studio is a VMM administrator or VMM delegated administrator for the relevant Hyper-V machines. If this account only has the delegated administrator role in VMM, the storage data is not listed in Studio during the host creation process.
   - The user account used for Studio integration must also be a member of the administrators local security group on each Hyper-V Server to support VM life cycle management (such as VM creation, update, and deletion).
   
   Note: Installing the Delivery Controller directly on a server running Hyper-V is not supported.

Create a Master VM

You create a master VM to provide user desktops.

1. Install the Virtual Desktop Agent on the Master VM, and make sure that you select the option to optimize the desktop. This improves the performance of users' desktops by reconfiguring various Windows features that are incompatible with or unnecessary for virtual desktops.
2. Take a snapshot of the Master VM to use as a back-up. For more information, see Preparing a Master image.

Create virtual desktops

If you are using Citrix Studio to create VMs, rather than selecting an existing Machine Catalog, run the Studio Deployment wizard and create virtual desktops as follows:

1. On the Host page, select Virtual Machines and then select Microsoft virtualization as the host type.
2. Enter the service address as the fully qualified domain name of the host server.
3. Enter the credentials for the administrator account you set up earlier that has permissions to create new VMs.
4. In the Host Details dialog box, select the cluster or standalone host to use when creating new VMs.
   
   Important: Browse for and select a cluster or standalone host even if you are using a single Hyper-V host deployment.

Support for Microsoft System Center Virtual Machine Manager

This release supports:

- VMM 2012 — Provides improved management capabilities, letting you manage the entire virtualized datacenter as well as virtual machines. This release now orchestrates cluster host patching as well as integrating with Windows Server Update Services, allowing you to define baselines of patches that each host needs.
- VMM 2012 SP1 — Provides performance improvements for Machine Creation Services (MCS) when using SMB 3.0 on file servers with clustered shared volumes and Storage Area Networks (SANs). These file shares provide low cost caching and reduced I/O on the SAN storage improving the performance.
- VMM 2012 R2 — Enables at-scale management of major Windows Server 2012 R2 capabilities, including running VM snapshots, dynamic VHDX resize, and Storage Spaces.
### Machine Creation Services (MCS) on SMB 3 file shares

For Machine Catalogs created through MCS on SMB 3 file shares for VM storage, make sure that credentials are set up as follows so that calls from a Delivery Controller’s Hypervisor Communications Library (HCL) can successfully connect to SMB storage:

- VMM user credentials must include full read write access to the SMB storage.
- Storage virtual disk operations during VM life cycle events are performed through the Hyper-V server using the VMM user credentials.

Note: If you use SMB as storage, enable the CredSSP from the Delivery Controller to individual Hyper-V machines when using VMM 2012 SP1 with Hyper-V on Windows Server 2012. For more information, see the KB article [http://support.citrix.com/article/CTX137465](http://support.citrix.com/article/CTX137465).

Using a standard PowerShell V3 Remote session, the HCL opens a connection to the Hyper-V machine using the Authentication Credential Security Support Provider (CredSSP) feature. This feature passes users’ credentials across to the Hyper-V Machine (Kerberos encrypted) and the PowerShell commands in this session on the remote Hyper-V machine run with the credentials provided (in this case, those of the VMM User), so that communication commands to storage correctly work.

The following tasks use PowerShell scripts that originate in the Delivery Controller HCL and are then sent to the Hyper-V machine to act on the SMB 3.0 storage.

### Consolidate Master Image

**Use**

A Master Image creates a new MCS Provisioning scheme (Machine Catalog). It clones and flattens the Master VM ready for creating new VMs from the new disk created (and removes dependency on the original master VM).

**PowerShell script operation**

**ConvertVirtualHardDisk** on the root\virtualization\v2 namespace

**Example**

``` Powershell
$ims = Get-WmiObject -class $class -namespace "root\virtualization\v2";
$result = $ims.ConvertVirtualHardDisk($diskName, $vhdastext)
$result
```

**Create difference disk**

**Use**

Creates a difference disk from the Master Image generated by consolidating the Master Image. The difference disk is then attached to a new VM.

**CreateVirtualHardDisk** on the root\virtualization\v2 namespace

**Example**

``` Powershell
$ims = Get-WmiObject -class $class -namespace "root\virtualization\v2";
$result = $ims.CreateVirtualHardDisk($vhdastext);
$result
```
Upload identity disks

Use

The Hypervisor Communications Library (HCL) cannot directly upload the identity disk to SMB storage. Therefore, the Hyper-V machine must upload and copy the identity disk to the storage. Because the Hyper-V machine cannot read the disk from the Delivery Controller, HCL must first copy the identity disk through the Hyper-V machine as follows.

Operation

1. HCL uploads the Identity to the Hyper-V machine through the administrator share.
2. Hyper-V machine copies the disk to the SMB storage through a PowerShell script running in the PowerShell V3 remote session. A folder is created on the Hyper-V machine and the permissions on that folder are locked for the VMM user only (through the remote PowerShell connection).
3. HCL deletes the file from the administrator share.
4. When the HCL completes the identity disk upload to the Hyper-V machine. The remote PowerShell session copies the identity disks to SMB storage and then deletes it from the Hyper-V machine.
   Note: The identity disk folder is recreated if it is deleted so that it is available for reuse.

Download identity disks

Use

As with uploads, the identity disks pass through the Hyper-V machine to the HCL. The following process creates a folder that only has VMM user permissions on the Hyper-V server if it does not exist.

Operation

1. The Hyper-V machine copies the disk from the SMB storage to local Hyper-V storage through a PowerShell script running in the PowerShell V3 remote session.
2. HCL reads the disk from the Hyper-V machine's administrator share into memory.
3. HCL deletes the file from the administrator share.

Personal vDisk creation

Use

If the administrator creates the VM in a Personal vDisk Machine Catalog, you must create an empty disk (Personal vDisk).

Operation

The call to create an empty disk does not require direct access to the storage. If you have PVD disks that reside on different storage than the Main or Operating System disk, then the use Remote PowerShell to create the Pvd disk in a directory folder that has the same name of the VM from which it was created. For CSV or LocalStorage, do not use Remote PowerShell. Creating the directory before creating an empty disk avoids VMM command failure.

From the Hyper-V machine, perform a mkdir on the storage.

For more information about using the SDK, see About the XenDesktop SDK.
Properties for Microsoft System Center Configuration Manager

Oct 17, 2013

Properties are available to Microsoft System Center configuration Manager (Configuration Manager) 2012 and 2012 R2 so that you can manage virtual desktops using this tool.

Note: Although Configuration Manager 2012 R2 provides the same functionality as Configuration Manager 2012 SP1, new features found in Configuration Manager 2012 R2 are not supported.

The properties are available for the **Citrix_virtualDesktopInfo** class in the Root\Citrix\DesktopInformation namespace.

The following properties are available. Property names come from the Windows Management Instrumentation (WMI) provider:

- **AssignmentType** – Sets the value of **IsAssigned**. The valid **AssignmentType** values are:
  - **ClientIP**
  - **ClientName**
  - **None**
  - **User** – This value sets **IsAssigned** to True
- **BrokerSiteName** – Site; returns the same value as **HostIdentifier**
- **DesktopCatalogName** – Machine Catalog associated with the desktop
- **DesktopGroupName** – Delivery group associated with the desktop
- **HostIdentifier** – Site; returns the same value as BrokerSiteName
- **IsAssigned** – True to assign the desktop to a user, set to False for a random desktop
- **IsMasterImage** – Allows decisions about the environment. For example, you may want to install applications on the Master Image and not on the provisioned machines, especially if those machines are in a clean state on boot machines. The values are:
  - **True** on a Virtual Machine (VM) that is used as a Master Image (This value is set during installation based on a selection during in the installation process).
  - **Cleared** on a VM that is provisioned from that image.
- **IsVirtualMachine** – True for a virtual machine, false for a physical machine.
- **OSChangesPersist** – False if the desktop operating system image is reset to a clean state every time it is restarted, otherwise true.
- **PersistentDataLocation** – The location where Configuration Manager stores persistent data. This is not accessible to users.
- **PersonalvDiskDriveLetter** – For a desktop with a Personal vDisk, the drive letter you assign to the Personal vDisk.

The properties **BrokerSiteName**, **DesktopCatalogName**, **DesktopGroupName**, and **HostIdentifier** are determined when the desktop registers with the controller they are null for a desktop that has not fully registered.

You can display the properties by using the hardware inventory in Configuration Manager, or by using attributes of
Configuration Manager objects. In these instances, the names may include spaces or vary slightly from the property names. For example, **BrokerSiteName** may appear as Broker Site Name.

You can also use Microsoft properties in the Microsoft class **CCM_DesktopMachine** in the Root\ccm_vdi namespace. For more information on these properties, see the Microsoft documentation.

Note: Boolean properties displayed in Configuration Manager 2012 may appear as 1 or 0, not true or false.
VMware and virtual machine management

May 02, 2014
Configure your system as described in this topic if you use VMware to provide virtual machines in your environment.

See the — System Requirements topic for supported VMware version information.

Install and configure your hypervisor

1. Install vCenter Server and the appropriate management tools required.
   Note: No support is provided for vSphere vCenter Linked Mode operation.
2. Create a VMware user account with the following permissions, at the DataCenter level, at a minimum:
   Note: This account has permissions to create new VMs and is used to communicate with vCenter.

<table>
<thead>
<tr>
<th>SDK</th>
<th>User Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datastore.AllocateSpace</td>
<td>Datastore &gt; Allocate space</td>
</tr>
<tr>
<td>Datastore.Browse</td>
<td>Datastore &gt; Browse datastore</td>
</tr>
<tr>
<td>Datastore.FileManagement</td>
<td>Datastore &gt; Low level file operations</td>
</tr>
<tr>
<td>Network.Assign</td>
<td>Network &gt; Assign network</td>
</tr>
<tr>
<td>Resource.AssignVMToPool</td>
<td>Resource &gt; Assign virtual machine to resource pool</td>
</tr>
<tr>
<td>System.Anonymous</td>
<td>Added automatically.</td>
</tr>
<tr>
<td>System.Read</td>
<td>Added automatically.</td>
</tr>
<tr>
<td>System.View</td>
<td>Added automatically.</td>
</tr>
<tr>
<td>Task.Create</td>
<td>Tasks &gt; Create task</td>
</tr>
<tr>
<td>VirtualMachine.Config.AddRemoveDevice</td>
<td>Virtual machine &gt; Configuration &gt; Add or remove device</td>
</tr>
<tr>
<td>VirtualMachine.Config.AddExistingDisk</td>
<td>Virtual machine &gt; Configuration &gt; Add existing disk</td>
</tr>
<tr>
<td>VirtualMachine.Config.AddNewDisk</td>
<td>Virtual machine &gt; Configuration &gt; Add new disk</td>
</tr>
<tr>
<td>SDK</td>
<td>User Interface</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>VirtualMachine.Config.CPUCount</td>
<td>Virtual machine &gt; Configuration &gt; Change CPU Count</td>
</tr>
<tr>
<td>VirtualMachine.Config.Memory</td>
<td>Virtual machine &gt; Configuration &gt; Memory</td>
</tr>
<tr>
<td>VirtualMachine.Config.RemoveDisk</td>
<td>Virtual machine &gt; Configuration &gt; Remove disk</td>
</tr>
<tr>
<td>VirtualMachine.Config.Settings</td>
<td>Virtual machine &gt; Configuration &gt; Settings</td>
</tr>
<tr>
<td>VirtualMachine.Interact.Reset</td>
<td>Virtual machine &gt; Interaction &gt; Reset</td>
</tr>
<tr>
<td>VirtualMachine.Inventory.Create</td>
<td>Virtual machine &gt; Inventory &gt; Create new</td>
</tr>
<tr>
<td>VirtualMachine.Inventory.CreateFromExisting</td>
<td>Virtual machine &gt; Inventory &gt; Create from existing</td>
</tr>
<tr>
<td>VirtualMachine.Inventory.Delete</td>
<td>Virtual machine &gt; Inventory &gt; Remove</td>
</tr>
<tr>
<td>VirtualMachine.Inventory.Register</td>
<td>Virtual machine &gt; Inventory &gt; Register</td>
</tr>
<tr>
<td>VirtualMachine.Provisioning.Clone</td>
<td>Virtual machine &gt; Provisioning &gt; Clone virtual machine</td>
</tr>
<tr>
<td>VirtualMachine.Provisioning.GetVmFiles</td>
<td>Virtual machine &gt; Provisioning &gt; Allow virtual machine download</td>
</tr>
<tr>
<td>VirtualMachine.Provisioning.PutVmFiles</td>
<td>Virtual machine &gt; Provisioning &gt; Allow virtual machine files upload</td>
</tr>
<tr>
<td>VirtualMachine.Provisioning.DeployTemplate</td>
<td>Virtual machine &gt; Provisioning &gt; Deploy template</td>
</tr>
</tbody>
</table>
3. If you want the VMs you create to be tagged, add the following permissions for the user account:

<table>
<thead>
<tr>
<th>SDK</th>
<th>User Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>VirtualMachine.Provisioning.MarkAsVM</td>
<td>Virtual machine &gt; Provisioning &gt; Mark as virtual machine</td>
</tr>
</tbody>
</table>
| VirtualMachine.State.CreateSnapshot     | • For vSphere 5.0, Update 2 and vSphere 5.1, Update 1: 
                                          |   Virtual machine > State > Create snapshot          |
|                                          | • For vSphere 5.5: 
                                          |   Virtual machine > Snapshot management > Create snapshot |
| VirtualMachine.State.RemoveSnapshot     | • For vSphere 5.0, Update 2 and vSphere 5.1, Update 1: 
                                          |   Virtual machine > State > Remove snapshot          |
|                                          | • For vSphere 5.5: 
                                          |   Virtual machine > Snapshot management > Remove snapshot |
| VirtualMachine.State.RevertToSnapshot   | • For vSphere 5.0, Update 2 and vSphere 5.1, Update 1: 
                                          |   Virtual machine > State > Revert to snapshot       |
|                                          | • For vSphere 5.5: 
                                          |   Virtual machine > Snapshot management > Revert to snapshot |

To ensure that you use a clean base image for creating new VMs, tag VMs created with Machine Creation Services to exclude them from the list of VMs available to use as base images.

4. To protect vSphere communications, Citrix recommends that you use HTTPS rather than HTTP. HTTPS requires digital certificates. Citrix recommends you use a digital certificate issued from a certificate authority in accordance with your organization's security policy.

If you are unable to use a digital certificate issued from a certificate authority, and your organization's security policy permits it, you can use the VMware-installed self-signed certificate. The VMware vCenter certificate needs to be added in each of the delivery controllers in your environment. To do this:

1. Add the fully qualified domain name (FQDN) of the computer running vCenter Server to the hosts file on that server, located at %SystemRoot%/WINDOWS/system32/Drivers/etc/. This step is required only if the FQDN of the computer running vCenter Server is not already present in the domain name system.
2. Obtain the vCenter certificate using any of the following methods:
   • Copy from the vCenter server:
     1. Copy the file rui.crt from the vCenter server to a location accessible on your delivery controllers. The default location on the vCenter server is c:\Documents and Settings\All Users\Applications Data\VMware\VMware VirtualCenter\SSL\
2. On your delivery controller, navigate to the location of the exported certificate and open the rui.crt file.

- Download the certificate using a web browser. If you are using Internet Explorer, depending on your user account, you may need to right-click on Internet Explorer and choose Run as Administrator to download or install the certificate.

1. Open your web browser and make a secure web connection to the vCenter server; for example https://server1.domain1.com
2. Accept the security warnings.
3. Click on the address bar where it shows the certificate error.
4. View the certificate and click on the Details tab.
5. Select Copy to file... and export in .CER format, providing a name when prompted to do so.
6. Save the exported certificate.
7. Navigate to the location of the exported certificate and open the .CER file.

- Import directly from Internet Explorer running as an administrator:

1. Open your web browser and make a secure web connection to the vCenter server; for example https://server1.domain1.com
2. Accept the security warnings.
3. Click on the address bar where it shows the certificate error.
4. View the certificate.

3. Import the certificate into the certificate store on each of your delivery controllers:

1. Click Install certificate, select Local Machine, and then click Next.
2. Select Place all certificates in the following store, and then click Browse.
   - If you are using Windows Server 2008 R2:
     1. Select the Show physical stores check box.
     2. Expand Trusted People.
     3. Select Local Computer.
     4. Click Next, then click Finish.
   - If you are using Windows Server 2012 or Windows Server 2012 R2:
     1. Select Trusted People, then click OK.
     2. Click Next, then click Finish.

   Important: If you change the name of the vSphere server after installation, you must generate a new self-signed certificate on that server before following the process to import the new certificate.

Create a master VM

You create a master VM to provide users' desktops and applications.

1. Install the VDA on the master VM, ensuring you select the option to optimize the desktop. This improves the performance of users' desktops and applications by reconfiguring various Windows features that are incompatible with or unnecessary for virtual desktops.
2. Take a snapshot of the master VM to use as a back-up. For more information, see Prepare a master image.

Create virtual desktops

If you are using Studio to create VMs, rather than selecting an existing machine catalog, ensure you enter the following information when setting up your hosting infrastructure to create virtual desktops:

1. On the Host Connection page, select VMWare vSphere® as the host type.
2. Enter the address of the access point for the vCenter SDK.
   - For example, https://vmware.example.com/sdk.
3. Enter the credentials for the VMware user account you set up earlier that has permissions to create new VMs. Ensure you specify the username for the account in the format Domain/Username.
There are several security considerations to consider when planning your deployment. See Security planning for planning and best practices information.

Smart cards offer a number of benefits to customers, ranging from convenience for end users to increase security for the infrastructure.

Pass-through authentication reduces the number of PIN entries. Pass-through authentication uses Kerberos instead of smart card authentication.

- Security planning
- Smart card system requirements
- Plan pass-through authentication with smart cards
Authenticate securely with smart cards

Mar 21, 2015

Considerations when managing smart cards in your organization:

- Multiple smart cards and multiple readers can be used on the same user device, but if pass-through authentication is in use, only one smart card must be inserted when the user starts a virtual desktop or application. When a smart card is used within an application (for example, for digital signing or encryption functions), there might be additional prompts to insert a smart card or enter a PIN. This can occur if more than one smart card has been inserted at the same time. If users are prompted to insert a smart card when the smart card is already in the reader, they should select Cancel. If they are prompted for the PIN, they should enter the PIN again.

- If you are using hosted applications running on Windows Server 2008 or 2008 R2 and with smart cards requiring the Microsoft Base Smart Card Cryptographic Service Provider, you might find that if a user runs a smart card transaction, all other users who use a smart card in the logon process are blocked. For further details and a hotfix for this issue, see http://support.microsoft.com/kb/949538.

Your organization might have specific security policies concerning the use of smart cards. These policies might, for example, state how smart cards are issued and how users should safeguard them. Some aspects of these policies might need to be reassessed in a XenDesktop environment.

You can reset PINs using a card management system or vendor utility.

Card readers

ZKA (Zentraler Kredit Ausschuss or Central Credit Committee) Class 1 contact card readers that comply with the USB Chip/Smart Card Interface Devices (CCID) specification are supported. These contain a slot or swipe into which the user inserts the smart card. Other classes, including Class 2 (readers with keypads for entering PINs), contactless readers, and virtual smart cards based on the Trusted Platform Module (TPM) chip, are not supported.

You must obtain a device driver for the smart card reader and install it on the user device. Many smart card readers can use the CCID device driver supplied by Microsoft.

You must obtain a device driver and cryptographic service provider (CSP) software from your smart card vendor, and install them on both user devices and virtual desktops. The driver and CSP software must be compatible with Citrix XenDesktop. Check your vendor's documentation for compatibility. Citrix recommends that you:

- Install the drivers and CSP software before installing any Citrix software on it.
- Install and test the drivers on a physical computer before installing Citrix software.

Note: For virtual desktops running Windows 7 using smart cards that support and use the mini driver model, smart card mini drivers should download automatically, but you can obtain them from http://catalog.update.microsoft.com or from your vendor. Additionally, if PKCS#11 middleware is required, obtain it from the card vendor.

Smart card support also involves components available from Citrix partners. These are updated independently by the partners, and are not described in these topics. For more information, refer to the Citrix Ready program at http://www.citrix.com/ready/.

Remote access with smart cards

- Smart cards are supported only for remote access to physical office PCs running Windows 7 or Windows 8; smart cards...
are not supported for office PCs running Windows XP.

- The following smart cards were tested with remote PC access:
  - Gemalto .Net 2.0 with the Gemalto .Net mini driver
  - Gemalto IDPrime .NET 510 with Gemalto .Net mini driver
  - Gemalto PIV cards with Actividentity ActivClient 6.2

**User devices**

User devices must run Citrix Receiver, appropriate middleware, and one of the following operating systems:

- Windows 8 (including Embedded Edition), 32-bit and 64-bit editions
- Windows 7 (including Embedded Edition), 32-bit and 64-bit editions
- Windows XP Professional, Service Pack 3 (32-bit edition)

**Middleware**

Receiver smart card support is based on Microsoft Personal Computer/Smart Card (PC/SC) standard specifications. A minimum requirement is that smart cards and smart card devices must be supported by the underlying Windows operating system and must be approved by the Microsoft Windows Hardware Quality Labs (WHQL) be used on computers running qualifying Windows operating systems. See [http://www.microsoft.com](http://www.microsoft.com) for additional information about hardware PC/SC compliance.

The following smart card and middleware combinations have been tested by Citrix as representative examples of their type. However, other smart cards and middleware can also be used. For more information about Citrix-compatible smart cards and middleware, see [http://www.citrix.com/ready](http://www.citrix.com/ready).

<table>
<thead>
<tr>
<th>Windows</th>
<th>Matching cards</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActivClient 6.2 (DoD CAC edition) in GSC-IS mode</td>
<td>DoD CAC card</td>
<td></td>
</tr>
<tr>
<td>ActivClient 7.0 in GSC-IS mode</td>
<td>DoD CAC card</td>
<td></td>
</tr>
<tr>
<td>ActivClient 7.0 in PIV mode</td>
<td>DoD CAC card, NIST PIV card</td>
<td></td>
</tr>
<tr>
<td>GemAlto Mini Driver for .NET card</td>
<td>GemAlto IDPrime .NET 510</td>
<td></td>
</tr>
<tr>
<td>SafeNet Authentication Client 8.x for Windows</td>
<td>USB eToken 5100</td>
<td></td>
</tr>
</tbody>
</table>

**Other requirements**

Carry out these additional tasks prior to your smart card deployment:

- Ensure that your public key infrastructure (PKI) is configured appropriately. This includes ensuring that certificate-to-account mapping is correctly configured for Active Directory environment and that user certificate validation can be performed successfully.
• Configure components to use TLS 1.0 for smart card logon.
• Ensure your deployment meets the system requirements of the other Citrix components used with smart cards, including Receiver and StoreFront.
To enable smart card usage

Mar 28, 2015

To enable smart card usage, you will need access to servers running the following associated with your Citrix XenDesktop installations:

- The Active Directory domain controller for the user account that is associated with a login certificate on the smart card
- Delivery Controller
- Citrix StoreFront
- Citrix NetScaler Gateway/Citrix Access Gateway 10.x
- Virtual Delivery Agent
- (Optional for remote access): Microsoft Exchange Server

Note: For Remote PC Access, smart cards are supported on physical computers running Windows 7 or Windows 8 only; smart cards are not supported for physical computers running Windows XP or Vista.

1. Familiarize yourself with smart card technology and your specific smart card technology.
2. Understand how to install and maintain certificates in distributed environments.
3. Familiarize yourself with the operation and documentation of the following:
   1. Citrix StoreFront 2.0
   2. Citrix NetScaler Gateway/Citrix Access Gateway 10.x
      Note: This is required for most smart card deployments.
   3. Citrix Receiver for Windows 4.0 and 3.4
   4. Citrix XenDesktop SDK
4. Enable XenDesktop for smart card use.
   1. Issue smart cards to the users according to your card issuance policy.
   2. (Optional) Set up smart card to enable users for Remote PC Access.
   3. Install and configure the Delivery Controller and StoreFront (if not already installed for smart card remoting).
5. Enable StoreFront for smart card use. For details, see the StoreFront product documentation.
6. Enable NetScaler Gateway/Access Gateway for smart card use. For details, see Configuring Authentication and Authorization and Configuring Smart Card Access with the Web Interface
7. Enable the Virtual Delivery Agent (VDA) for smart card use. See Install using the graphical interface for details about installing and configuring the Virtual Delivery Agent.
   1. Ensure the Virtual Delivery Agent has the required applications and updates.
   2. Install the middleware.
   3. Set up smart card remoting, enabling the communication of smart card data between Receiver on a user device and a virtual desktop session.
8. Enable user devices for smart card use. See the StoreFront product documentation for details.

Windows user devices

These user devices include domain-joined or non-domain-joined computers.

1. Import the certificate authority root certificate and the issuing certificate authority certificate into the device's keystore.
2. Install your vendor's cryptographic middleware.
3. Install and configure Receiver for Windows, being sure to import icaclient.adm using the Group Policy Management Console and enabling smart card authentication.
   - For thin clients and computers running Desktop Lock, install Receiver for Windows Enterprise 3.4.
9. Test your deployment.
   Ensure that your deployment is configured correctly by launching a virtual desktop with a test user's smart card. Test all possible access mechanisms (for example, accessing the desktop through Internet Explorer and Receiver).
Supported smart card deployments

Jun 18, 2013

The following types of smart card deployments are supported by this release and by mixed environments involving this release and Citrix XenDesktop 5.6 Feature Pack 1. The deployments are described in detail in other topics in this section. Other configurations may work but are not supported.

<table>
<thead>
<tr>
<th>Type</th>
<th>StoreFront connectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local domain-joined computers</td>
<td>Directly connected</td>
</tr>
<tr>
<td>Remote access from domain-joined computers</td>
<td>Connected through NetScaler Gateway</td>
</tr>
<tr>
<td>Non-domain-joined computers</td>
<td>Directly connected</td>
</tr>
<tr>
<td>Remote access from non-domain-joined computers</td>
<td>Connected through NetScaler Gateway</td>
</tr>
<tr>
<td>Non-domain-joined computers and thin clients accessing the Desktop Appliance site</td>
<td>Connected through Desktop Appliance sites</td>
</tr>
<tr>
<td>Domain-joined computers and thin clients accessing StoreFront through the XenApp Services URL</td>
<td>Connected through XenApp Services URLs</td>
</tr>
</tbody>
</table>

The deployment types are defined by these characteristics of the user device to which the smart card reader is connected:

- Whether the device is domain-joined or non-domain-joined.
- How the device is connected to StoreFront.
- What software is used to view virtual desktops and applications.

In addition, smart card-enabled applications, such as Microsoft Word, and Microsoft Excel, can also be used in these deployments. These applications allow users to digitally sign or encrypt documents.

Bimodal authentication

Where possible in each of these deployments, Receiver supports bimodal authentication by offering the user a choice between using a smart card and entering their user name and password. This is useful if the smart card cannot be used (for example, the user has left it at home or the logon certificate has expired).

Because users of non-domain-joined devices log on to Receiver for Windows directly, you can enable users to fall back to explicit authentication. If you configure bimodal authentication, users are initially prompted to log on using their smart cards and PINs but have the option to select explicit authentication if they experience any issues with their smart cards.

If you deploy NetScaler Gateway, users log on to their devices and are prompted by Receiver for Windows to authenticate to NetScaler Gateway. This applies to both domain-joined and non-domain-joined devices. Users can log on to NetScaler Gateway using either their smart cards and PINs, or with explicit credentials. This enables you to provide users with bimodal authentication for NetScaler Gateway logons. Configure pass-through authentication from NetScaler Gateway to StoreFront and delegate credential validation to NetScaler Gateway for smart card users so that users are silently...
Multiple Active Directory forest considerations

In a Citrix environment, smart cards are supported within a single forest. Smart card logons across forests require a direct two-way forest trust to all user accounts. More complex multi-forest deployments involving smart cards (that is, where trusts are only one-way or of different types) are not supported.

You can use smart cards in a Citrix environment that includes remote desktops. This feature can be installed locally (on the user device that the smart card is connected to) or remotely (on the remote desktop that the user device connects to).

Smart card removal policy

The smart card removal policy set on XenDesktop determines what happens if you remove the smart card from the reader during a session. The smart card removal policy is configured through and handled by the Windows operating system.

<table>
<thead>
<tr>
<th>Policy setting</th>
<th>Desktop behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>No action</td>
<td>No action.</td>
</tr>
<tr>
<td>Lock workstation</td>
<td>The desktop session is disconnected and the virtual desktop is locked.</td>
</tr>
<tr>
<td>Force logoff</td>
<td>The user is forced to log off. If the network connection is lost and this setting is enabled, the session may be logged off and the user may lose data.</td>
</tr>
<tr>
<td>Disconnect if a remote Terminal Services session</td>
<td>The session is disconnected and the virtual desktop is locked.</td>
</tr>
</tbody>
</table>

Certificate revocation checking

If certificate revocation checking is enabled and a user inserts a smart card with an invalid certificate into a card reader, the user cannot authenticate or access the desktop or application related to the certificate. For example, if the invalid certificate is used for email decryption, the email remains encrypted. If other certificates on the card, such as ones used for authentication, are still valid, those functions remain active.
Smart card deployment for domain-joined computers

Nov 01, 2013
This deployment involves domain-joined user devices that run the Desktop Viewer and connect directly to StoreFront.

A user logs on to a device using a smart card and PIN. Receiver authenticates the user to a Storefront server using Integrated Windows Authentication (IWA). StoreFront passes the user security identifiers (SIDs) to Citrix XenDesktop. When the user starts a virtual desktop or application, the user is not prompted for a PIN again because the single sign-on feature is configured on Receiver.

This deployment can be extended to a double-hop with the addition of a second StoreFront server and a server hosting applications. A Receiver from the virtual desktop authenticates to the second StoreFront server. Any authentication method can be used for this second connection. The configuration shown for the first hop can be reused in the second hop or used in the second hop only.
Smart card deployment for remote access from domain-joined computers

Nov 01, 2013

This deployment involves domain-joined user devices that run the Desktop Viewer and connect to StoreFront through NetScaler Gateway/Access Gateway.

A user logs on to a device using a smart card and PIN, and then logs on again to NetScaler Gateway/Access Gateway. This second logon can be with either the smart card and PIN or a user name and password because Receiver allows bimodal authentication in this deployment.

The user is automatically logged on to StoreFront, which passes the user security identifiers (SIDs) to Citrix XenDesktop. When the user starts a virtual desktop or application, the user is not prompted again for a PIN because the single sign-on feature is configured on Receiver.

This deployment can be extended to a double-hop with the addition of a second StoreFront server and a server hosting applications. A Receiver from the virtual desktop authenticates to the second StoreFront server. Any authentication method can be used for this second connection. The configuration shown for the first hop can be reused in the second hop or used in the second hop only.
Smart card deployment for non-domain-joined computers

Nov 01, 2013

This deployment involves non-domain-joined user devices that run the Desktop Viewer and connect directly to StoreFront.

A user logs on to a device. Typically, the user enters a user name and password but, since the device is not joined to a domain, credentials for this logon are optional. Because bimodal authentication is possible in this deployment, Receiver prompts the user either for a smart card and PIN or a user name and password. Receiver then authenticates to Storefront.

StoreFront passes the user security identifiers (SIDs) to Citrix XenDesktop. When the user starts a virtual desktop or application, the user is prompted for a PIN again because the single sign-on feature is not available in this deployment.

This deployment can be extended to a double-hop with the addition of a second StoreFront server and a server hosting applications. A Receiver from the virtual desktop authenticates to the second StoreFront server. Any authentication method can be used for this second connection. The configuration shown for the first hop can be reused in the second hop or used in the second hop only.
Smart card deployment for remote access from non-domain-joined computers

Nov 01, 2013

This deployment involves non-domain-joined user devices that run the Desktop Viewer and connect directly to StoreFront.

A user logs on to a device. Typically, the user enters a user name and password but, since the device is not joined to a domain, credentials for this logon are optional. Because bimodal authentication is possible in this deployment, Receiver prompts the user either for a smart card and PIN or a user name and password. Receiver then authenticates to Storefront.

StoreFront passes the user security identifiers (SIDs) to Citrix XenDesktop. When the user starts a virtual desktop or application, the user is prompted for a PIN again because the single sign-on feature is not available in this deployment.

This deployment can be extended to a double-hop with the addition of a second StoreFront server and a server hosting applications. A Receiver from the virtual desktop authenticates to the second StoreFront server. Any authentication method can be used for this second connection. The configuration shown for the first hop can be reused in the second hop or used in the second hop only.
Smart card deployment for non-domain-joined computers and thin clients accessing the Desktop Appliance site

Nov 01, 2013
This deployment involves non-domain-joined user devices that may run the Desktop Lock and connect to StoreFront through Desktop Appliance sites.

The Desktop Lock is a separate component that is released with Citrix XenDesktop and Citrix VDI-in-a-Box. It is an alternative to the Desktop Viewer and is designed mainly for repurposed Windows computers and Windows thin clients. The Desktop Lock replaces the Windows shell and Task Manager in these user devices, preventing users from accessing the underlying devices. With the Desktop Lock, users can access Windows Server Machine desktops and Windows Desktop Machine desktops.

Note: Installation of Desktop Lock is optional.

A user logs on to a device with a smart card. If Desktop Lock is running on the device, the device is configured to launch a Desktop Appliance site through Internet Explorer running in Kiosk Mode. An ActiveX control on the site prompts the user for a PIN, and sends it to StoreFront. StoreFront passes the user security identifiers (SIDs) to Citrix XenDesktop. The first available desktop in the alphabetical list in an assigned Desktop Group starts.

This deployment can be extended to a double-hop with the addition of a second StoreFront server and a server hosting applications. A Receiver from the virtual desktop authenticates to the second StoreFront server. Any authentication method can be used for this second connection. The configuration shown for the first hop can be reused in the second hop or used in the second hop only.
Smart card deployment for domain-joined computers and thin clients accessing StoreFront through the XenApp Services URL

Nov 01, 2013
This deployment involves domain-joined user devices that run the Desktop Lock and connect to StoreFront through XenApp Services URLs.

The Desktop Lock is a separate component that is released with Citrix XenDesktop and Citrix VDI-in-a-Box. It is an alternative to the Desktop Viewer and is designed mainly for repurposed Windows computers and Windows thin clients. The Desktop Lock replaces the Windows shell and Task Manager in these user devices, preventing users from accessing the underlying devices. With the Desktop Lock, users can access Windows Server Machine desktops and Windows Desktop Machine desktops.

Note: Installation of Desktop Lock is optional

A user logs on to a device using a smart card and PIN. If Desktop Lock is running on the device, it authenticates the user to a Storefront server using Integrated Windows Authentication (IWA). StoreFront passes the user security identifiers (SIDs) to Citrix XenDesktop. When the user starts a virtual desktop, the user is not prompted for a PIN again because the single sign-on feature is configured on Receiver.

This deployment can be extended to a double-hop with the addition of a second StoreFront server and a server hosting applications. A Receiver from the virtual desktop authenticates to the second StoreFront server. Any authentication method can be used for this second connection. The configuration shown for the first hop can be reused in the second hop or used in the second hop only.
Plan pass-through authentication with smart cards

Mar 28, 2015
Pass-through authentication with smart cards to virtual desktops is supported on user devices running the following operating systems:

- Windows 8
- Windows 7 SP1, Enterprise and Professional Editions
- Windows XP Service Pack 3, 32-bit edition

Pass-through authentication with smart cards to hosted applications is supported on servers running the following operating systems:

- Windows Server 2012
- Windows Server 2008

To use pass-through authentication with smart cards hosted applications, ensure you enable the use of Kerberos when you configure Pass-through with smartcard as the authentication method for the site.

Note: The availability of pass-through authentication with smart cards is dependent upon many factors including, but not limited to:
- Your organization's security policies regarding pass-through authentication.
- Middleware type and configuration.
- Smart card reader types.
- Middleware PIN caching policy.

Pass-through authentication with smart cards is configured on Citrix StoreFront. See the StoreFront product documentation for details.

Single sign-on with smart cards

Single sign-on is a Citrix feature that implements pass-through authentication with virtual desktop and application launches. You can use this feature in domain-joined, direct-to-StoreFront and domain-joined, NetScaler-to-StoreFront smart card deployments to reduce the number of times that users enter their PIN. To use single sign-on in these deployment types, edit the following parameters in default.ica. This file is located on the StoreFront server:

- **Domain-joined, direct-to-StoreFront smart card deployments**—Set DisableCtrlAltDel to Off
- **Domain-joined, NetScaler-to-StoreFront smart card deployments**—Set UseLocalUserAndPassword to On

For more instructions on setting these parameters, see the StoreFront or NetScaler Gateway documentation.

The availability of single sign-on functionality is dependent upon many factors including, but not limited to:

- Your organization's security policies regarding single sign-on.
- Middleware type and configuration.
- Smart card reader types.
- Middleware PIN caching policy.

Note: When the user logs on to the Virtual Delivery Agent (VDA) when a smart card reader is attached, a Windows tile may appear representing the previous successful mode of authentication, such as smart card or password. As a result, when
single sign-on is enabled, the single sign-on tile may appear. To log on, the user must click Switch Users to select another tile as the single sign-on tile will not work.
Managing printers in your environment is a multistage process:

1. Plan your printing architecture. This includes analyzing your business needs, your existing printing infrastructure, how your users and applications interact with printing today, and which printing management model best applies to your environment.

2. Configure your printing environment, including creating the policies necessary to deploy your printing design.

3. Test a pilot printing configuration before deploying it to users.

4. Maintain your Citrix printing environment, including updating policies when new employees or servers are added.

5. Troubleshoot issues that may arise in your printing environment.

Before you begin planning your deployment, make sure that you understand these core concepts for printing:

- The types of printer provisioning available.
- How print jobs are routed.
- The basics of printer driver management.

Printing concepts build on Windows printing concepts. To configure and successfully manage printing in your environment, you must understand how Windows network and client printing works and how this translates into printing behavior in this environment.

Print process

In this environment, all printing is initiated (by the user) on machines hosting applications. Print jobs are redirected through the network print server or user device to the printing device.

There is no persistent workspace for users of virtual desktops and applications. When a session ends the user's workspace is deleted, thus all settings need to be rebuilt at the beginning of each session. As a result, each time a user starts a new session, the system must rebuild the user's workspace.

When a user prints:

- Determines what printers to provide to the user. This is known as printer provisioning.
- Restores the user's printing preferences.
- Determines which printer is the default for the session.

You can customize how to perform these tasks by configuring options for printer provisioning, print job routing, printer property retention, and driver management. Be sure to evaluate how the various option settings might change the performance of printing in your environment and the user experience.

Printer provisioning

The process that makes printers available in a session is known as provisioning. Printer provisioning is typically handled dynamically. That is, the printers that appear in a session are not predetermined and stored. Instead, the printers are assembled, based on policies, as the session is built during log on and reconnection. As a result, the printers can change according to policy, user location, and network changes, provided they are reflected in policies. Thus, users who roam to a different location might see changes to their workspace.

The system also monitors client-side printers and dynamically adjusts in-session auto-created printers based on additions,
deletions, and changes to the client-side printers. This dynamic printer discovery benefits mobile users as they connect from various devices.

This section describes the most common methods of printer provisioning.

**Universal Print Server**

The Citrix Universal Print Server provides universal printing support for network printers. The Universal Print Server uses the Universal print driver. This solution enables you to use a single driver on a Server OS machine to allow network printing from any device.

Citrix recommends the Citrix Universal Print Server for remote print server scenarios. The Universal Print Server transfers the print job over the network in an optimized and compressed format, thus minimizing network use and improving the user experience.

The Universal Print Server feature comprises:

- A client component, UPClient.
  Enable the UPClient on each Server OS machine that provisions session network printers and uses the Universal print driver.

- A server component, UPServer.
  Install UPServer on each print server that provisions session network printers and uses the Universal print driver for the session printers (whether or not the session printers are centrally provisioned).

For Universal Print Server requirements and setup details, refer to the system requirements and installation topics.

Note: The Universal Print Server is also supported for VDI-in-a-Box 5.3. For information about installing Universal Print Server with VDI-in-a-Box, refer to the VDI-in-a-Box installation topics in eDocs.

The following illustration shows the typical workflow for a network based printer in an environment that uses Universal Print Server.

When you enable the Citrix Universal Print Server, all connected network printers leverage it automatically through auto-discovery.

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.267
Autocreation

Autocreation refers to printers automatically created at the beginning of each session. Both remote network printers and locally attached client printers can be auto-created. Consider auto-creating only the default client printer for environments with a large number of printers per user. Auto-creating a smaller number of printers uses less overhead (memory and CPU) on Server OS machines. Minimizing auto-created printers can also reduce user logon times.

Auto-created printers are based on:

- The printers installed on the user device.
- Any policies that apply to the session.
  Autocreation policy settings enable you to limit the number or type of printers that are auto-created. By default, the printers are available in sessions when configuring all printers on the user device automatically, including locally attached and network printers.

After the user ends the session, the printers for that session are deleted.

Client and network printer autocreation has associated maintenance. For example, adding a printer requires that you:

- Update the Session printers policy setting.
- Add the driver to all Server OS machines using the Printer driver mapping and compatibility policy setting.

Print job routing

The term printing pathway encompasses both the path by which print jobs are routed and the location where print jobs are spooled. Both aspects of this concept are important. Routing affects network traffic. Spooling affects utilization of local resources on the device that processes the job.

In this environment, print jobs can take two paths to a printing device: Through the client or through a network print server. Those paths are referred to as the client printing pathway and the network printing pathway. Which path is chosen by default depends on the kind of printer used.

Locally attached printers

The system routes jobs to locally attached printers from the Server OS machine, through the client, and then to the print device. The ICA protocol optimizes and compresses the print job traffic. When a printing device is attached locally to the user device, print jobs are routed over the ICA virtual channel.
By default, all print jobs destined for network printers route from the Server OS machine, across the network, and directly to the print server. However, print jobs are automatically routed over the ICA connection in the following situations:

- If the virtual desktop or application cannot contact the print server.
- If the native printer driver is not available on the Server OS machine.

If the Universal Print Server is not enabled, configuring the client printing pathway for network printing is useful for low bandwidth connections, such as wide area networks, that can benefit from the optimization and traffic compression that results from sending jobs over the ICA connection.

The client printing pathway also lets you limit traffic or restrict bandwidth allocated for print jobs. If routing jobs through the user device is not possible, such as for thin clients without printing capabilities, Quality of Service should be configured to prioritize ICA/HDX traffic and ensure a good in-session user experience.

Print driver management

To simplify printing in this environments, Citrix recommends using Citrix Universal print driver. The Universal print driver is a device-independent driver that supports any print device and thus simplifies administration by reducing the number of drivers required. The Universal print driver supports advanced printer functionality, such as stapling and sorting, and does not limit color depth.

The following illustration shows the Universal print driver components and a typical workflow for a printer locally attached to a device.
When planning your driver management strategy, determine if you will support the Universal print driver, device-specific drivers, or both. If you support standard drivers, you need to determine:

- The types of drivers to support.
- Whether to install printer drivers automatically when they are missing from Server OS machines.
- Whether to create driver compatibility lists.

During printer autocreation, if the system detects a new local printer connected to a user device, it checks the Server OS machine for the required printer driver. By default, if a Windows-native driver is not available, the system uses the Universal print driver.

The printer driver on the Server OS machine and the driver on the user device must match for printing to succeed. The illustration that follows shows how a printer driver is used in two places for client printing.
Plan your printing configuration

Mar 21, 2015

Choosing the most appropriate printing configuration options for your needs and environment can simplify administration. Although the default print configuration enables users to print in most environments, the defaults might not provide the expected user experience or the optimum network usage and management overhead for your environment.

Your printing configuration depends upon:

- Your business needs and your existing printing infrastructure.
  Design your printing configuration around the needs of your organization. Your existing printing implementation (whether users can add printers, which users have access to what printers, and so on) might be a useful guide when defining your printing configuration.

- Whether your organization has security policies that reserve printers for certain users (for example, printers for Human Resources or payroll).

- Whether users need to print while away from their primary work location, such as workers who move between workstations or travel on business.

When designing your printing configuration, try to give users the same experience in a session as they have when printing from local user devices.

Example print deployment

The following illustration shows the print deployment for these use cases:

- **Branch A** – A small overseas branch office with a few Windows workstations. Every user workstation has a locally attached, private printer.
- **Branch B** – A large branch office with thin clients and Windows-based workstations. For increased efficiency, the users of this branch share network-based printers (one per floor). Windows-based print servers located within the branch manage the print queues.
- **Home office** – A home office with a Mac OS-based user device that accesses the company's Citrix infrastructure. The user device has a locally attached printer.
The following sections describe the configurations which minimize the complexity of the environment and simplify its management.

**Auto-created client printers and Citrix Universal printer driver**

In Branch A, all users work on Windows-based workstations, therefore auto-created client printers and the Universal printer driver are used. Those technologies provide these benefits:

- **Performance** – Print jobs are delivered over the ICA printing channel, thus the print data can be compressed to save bandwidth.
  
  To ensure that a single user printing a large document cannot degrade the session performance of other users, a Citrix policy is configured to specify the maximum printing bandwidth.

  An alternative solution is to leverage a multi-stream ICA connection, in which the print traffic is transferred within a separate low priority TCP connection. Multi-stream ICA is an option when Quality of Service (QoS) is not implemented on the WAN connection.

- **Flexibility** – Use of the Citrix Universal printer driver ensures that all printers connected to a client can also be used from a virtual desktop or application session without integrating a new printer driver in the data center.

**Citrix Universal Print Server**
In Branch B, all printers are network-based and their queues are managed on a Windows print server, thus the Citrix Universal Print Server is the most efficient configuration.

All required printer drivers are installed and managed on the print server by local administrators. Mapping the printers into the virtual desktop or application session works as follows:

- For Windows-based workstations – The local IT team helps users connect the appropriate network-based printer to their Windows workstations. This enables users to print from locally-installed applications. During a virtual desktop or application session, the printers configured locally are enumerated through autocreation. The virtual desktop or application then connects to the print server as a direct network connection if possible.

  The Citrix Universal Print Server components are installed and enabled, thus native printer drivers are not required. If a driver is updated or a printer queue is modified, no additional configuration is required in the data center.

- For thin clients – For thin client users, printers must be connected within the virtual desktop or application session. To provide users with the simplest printing experience, administrators configure a single Citrix Session Printer policy per floor to connect a floor's printer as the default printer. To ensure the correct printer is connected even if users roam between floors, the policies are filtered based on the subnet or the name of the thin client. That configuration, referred to as proximity printing, allows for local printer driver maintenance (according to the delegated administration model).

  If a printer queue needs to be modified or added, Citrix administrators must modify the respective Session printer policy within the environment.

Because the network printing traffic will be sent outside the ICA virtual channel, QoS is implemented. Inbound and outbound network traffic on ports used by ICA/HDX traffic are prioritized over all other network traffic. That configuration ensures that user sessions are not impacted by large print jobs.

**Auto-created client printers and Citrix Universal printer driver**

For home offices where users work on non-standard workstations and use non-managed print devices, the simplest approach is to use auto-created client printers and the Universal printer driver.

**Deployment summary**

In summary, the sample deployment is configured as follows:

- No printer drivers are installed on Server OS machines. Only the Citrix Universal printer driver is used. Fallback to native printing and the automatic installation of printer drivers are disabled.
- A policy is configured to auto-create all client printers for all users. Server OS machines will directly connect to the print servers by default. The only configuration required is to enable the Universal Print Server components.
- A session printer policy is configured for every floor of Branch B and applied to all thin clients of the respective floor.
- QoS is implemented for Branch B to ensure excellent user experience.

**Security considerations for printing**

Citrix printing solutions are secure by design.

- The Citrix Print Manager Service constantly monitors and responds to session events such as logon and logoff, disconnect, reconnect, and session termination. It handles service requests by impersonating the actual session user.
- Citrix printing assigns each printer a unique namespace in a session.
Citrix printing sets the default security descriptor for auto-created printers to ensure that client printers auto-created in one session are inaccessible to users running in other sessions. By default, administrative users cannot accidentally print to another session's client printer, even though they can see and manually adjust permissions for any client printer.
Best practices

Jun 18, 2013

Many factors determine the best printing solution for a particular environment. Some of these best practices might not apply to your site.

- Use the Citrix Universal Print Server.
- Use the Universal printer driver or Windows-native drivers.
- Minimize the number of printer drivers installed on Server OS machines.
- Use driver mapping to native drivers.
- Never install untested printer drivers on a production site.
- Avoid updating a driver. Always attempt to uninstall a driver, restart the print server, and then install the replacement driver.
- Uninstall unused drivers or use the Printer driver mapping and compatibility policy to prevent printers from being created with the driver.
- Try to avoid using version 2 kernel-mode drivers.
- To determine if a printer model is supported, contact the manufacturer or see the Citrix Ready product guide at www.citrix.com/ready.

In general, all of the Microsoft-supplied printer drivers are tested with Terminal Services and guaranteed to work with Citrix. However, before using a third-party printer driver, consult your printer driver vendor to ensure the driver is certified for Terminal Services by the Windows Hardware Quality Labs (WHQL) program. Citrix does not certify printer drivers.
Default print operations

Jun 18, 2013
By default, if you do not configure any policy rules, printing behavior is as follows:

- The Universal Print Server is disabled.
- All printers configured on the user device are created automatically at the beginning of each session. This behavior is equivalent to configuring the Citrix policy setting Auto-create client printers with the Auto-create all client printers option.
- The system routes all print jobs queued to printers locally attached to user devices as client print jobs (that is, over the ICA channel and through the user device).
- The system routes all print jobs queued to network printers directly from Server OS machines. If the system cannot route the jobs over the network, it will route them through the user device as a redirected client print job. This behavior is equivalent to disabling the Citrix policy setting Direct connection to print servers.
- The system attempts to store printing properties, a combination of the user's printing preferences and printing device-specific settings, on the user device. If the client does not support this operation, the system stores printing properties in user profiles on the Server OS machine. This behavior is equivalent to configuring the Citrix policy setting Printer properties retention with the Held in profile only if not saved on client option.
- The system uses the Windows version of the printer driver if it is available on the Server OS machine. If the printer driver is not available, the system attempts to install the driver from the Windows operating system. If the driver is not available in Windows, it uses a Citrix Universal print driver. This behavior is equivalent to enabling the Citrix policy setting Automatic installation of in-box printer drivers and configuring the Universal printing setting with the Use universal printing only if requested driver is unavailable.

Note: If you are unsure about what the shipping defaults are for printing, display them by creating a new policy and setting all printing policy rules to Enabled. The option that appears is the default.
Print policies and preferences

Jun 18, 2013

When users access printers from published applications, you can configure Citrix policies to specify:

- How printers are provisioned (or added to sessions)
- How print jobs are routed
- How printer drivers are managed

You can have different printing configurations for different user devices, users, or any other objects on which policies are filtered.

Most printing functions are configured through the Citrix Printing policies. Printing settings follow standard Citrix policy behavior.

The system can write printer settings to the printer object at the end of a session or to a client printing device, provided the user's network account has sufficient permissions. By default, Receiver uses the settings stored in the printer object in the session, before looking in other locations for settings and preferences.

By default, the system stores, or retains, printer properties on the user device (if supported by the device) or in the user profile on the Server OS machine. When a user changes printer properties during a session, those changes are updated in the user profile on the machine. The next time the user logs on or reconnects, the user device inherits those retained settings. That is, printer property changes on the user device do not impact the current session until after the user logs off and then logs on again.

Printing preferences

General locations of printing preferences

In Windows printing environments, changes made to printing preferences can be stored on the local computer or in a document. In this environment, when users modify printing settings, the settings are stored in these locations:

- **On the user device itself** – Windows users can change device settings on the user device by right-clicking the printer in the Control Panel and selecting Printing Preferences. For example, if Landscape is selected as page orientation, landscape is saved as the default page orientation preference for that printer.
- **Inside of a document** – In word-processing and desktop-publishing programs, document settings, such as page orientation, are often stored inside documents. For example, when you queue a document to print, Microsoft Word typically stores the printing preferences you specified, such as page orientation and the printer name, inside the document. These settings appear by default the next time you print that document.
- **From changes a user made during a session** – The system keeps only changes to the printing settings of an auto-created printer if the change was made in the Control Panel in the session; that is, on the Server OS machine.
- **On the Server OS machine** – These are the default settings associated with a particular printer driver on the machine.

The settings preserved in any Windows-based environment vary according to where the user made the changes. This also means that the printing settings that appear in one place, such as in a spreadsheet program, can be different than those in others, such as documents. As result, printing settings applied to a specific printer can change throughout a session.

Hierarchy of user printing preferences
Because printing preferences can be stored in multiple places, the system processes them according to a specific priority. Also, it is important to note that device settings are treated distinctly from, and usually take precedence over, document settings.

By default, the system always applies any printing settings a user modified during a session (that is, the retained settings) before considering any other settings. When the user prints, the system merges and applies the default printer settings stored on the Server OS machine with any retained or client printer settings.

Saving user printing preferences

Citrix recommends that you do not change where the printer properties are stored. The default setting, which saves the printer properties on the user device, is the easiest way to ensure consistent printing properties. If the system is unable to save properties on the user device, it automatically falls back to the user profile on the Server OS machine.

Review the Printer properties retention policy setting if these scenarios apply:

- If you use legacy plug-ins that do not allow users to store printer properties on a user device.
- If you use mandatory profiles on your Windows network and want to retain the user's printer properties.
Provision printers

Jun 18, 2013

This section provides configuration information for the following printer provisioning methods:

- **Citrix Universal Print Server**
  The Citrix Universal Print Server provides universal printing support for network printers. The Universal Print Server uses the Universal print driver, a single driver on the Server OS machine that allows local or network printing from any device, including thin clients and tablets.

  The Universal Print Server and the Universal print driver are installed. To use the Universal Print Server, install it on your print servers, as described in the installation topics, and configure it.

- **Auto-created client printers**
  These universal printing solutions are provided for client printers:

  - **Citrix Universal Printer** – A generic printer created at the beginning of sessions that is not tied to a printing device. The Citrix Universal Printer is not required to enumerate the available client printers during logon, which can greatly reduce resource usage and decrease user logon times. The Universal Printer can print to any client-side printing device. The Citrix Universal Printer might not work for all user devices or Receivers in your environment. The Citrix Universal Printer requires a Windows environment and does not support the Citrix Offline Plug-in or applications that are streamed to the client. Consider using auto-created client printers and the Universal print driver for such environments.

  To use a universal printing solution for non-Windows Receivers, use one of the other Universal print drivers that are based on postscript/PCL and installed automatically.

  - **Citrix Universal print drivers** – A device-independent printer driver that supports advanced printer functionality, such as stapling and sorting. If you configure a Citrix Universal print driver, the system uses the EMF-based Universal print driver by default.

    The Citrix Universal print driver might create smaller print jobs than older or less advanced printer drivers. However, a device-specific driver might be needed to optimize print jobs for a specialized printer.

- **Administrator-assigned session printers**
  By default, network printers on the user device are created automatically at the beginning of sessions. The system enables you to reduce the number of network printers that are enumerated and mapped by specifying the network printers to be created within each session. Such printers are referred to as session printers.

  You can filter session printer policies by IP address to provide proximity printing. Proximity printing enables users within a specified IP address range to automatically access the network printing devices that exist within that same range. Proximity printing is provided by the Citrix Universal Print Server and does not require the configuration described in this section.

  Proximity printing might involve the following scenario:

  - The internal company network operates with a DHCP server which automatically designates IP addresses to users.
  - All departments within the company have unique designated IP address ranges.
  - Network printers exist within each department’s IP address range.

  When proximity printing is configured and an employee travels from one department to another, no additional printing device configuration is required. Once the user device is recognized within the new department’s IP address range, it will
have access to all network printers within that range.
Configure the Universal Print Server

When determining the best print solution for your environment, consider the following:

- The Universal Print Server provides features not available for the Windows Print Provider: Image and font caching, advanced compression, optimization, and QoS support.
- The Universal print driver supports the public device-independent settings defined by Microsoft. If users need access to device settings that are specific to a print driver manufacturer, the Universal Print Server paired with a Windows-native driver might be the best solution. With that configuration, you retain the benefits of the Universal Print Server while providing users access to specialized printer functionality. A trade-off to consider is that Windows-native drivers require maintenance.

To use the Universal Print Server with a Windows-native driver, enable the Universal Print Server. By default, if the Windows-native driver is available, it is used. Otherwise, the Universal print driver is used. To specify changes to that behavior, such as to use only the Windows-native driver or only the Universal print driver, update the Universal print driver usage policy setting.

Use the following Citrix policy settings to configure the Universal Print Server. For more information, refer to the on-screen policy settings help.

- **Universal Print Server enable.** Universal Print Server is disabled by default. When you enable Universal Print Server, you choose whether to use the Windows Print Provider if the Universal Print Server is unavailable. After you enable the Universal Print Server, a user can add and enumerate network printers through the Windows Print Provider and Citrix Provider interfaces.
- **Universal Print Server print data stream (CGP) port.** Specifies the TCP port number used by the Universal Print Server print data stream CGP (Common Gateway Protocol) listener. Defaults to 7229.
- **Universal Print Server web service (HTTP/SOAP) port.** Specifies the TCP port number used by the Universal Print Server listener for incoming HTTP/SOAP requests. Defaults to 8080.
- **Universal Print Server print stream input bandwidth limit (kbps).** Specifies the upper bound (in kilobits-per-second) for the transfer rate of print data delivered from each print job to the Universal Print Server using CGP. Defaults to 0 (unlimited).

The Universal Print Server honors other Citrix printing policy settings and interacts with them as noted in the following table. The information provided assumes that the Universal Print Server policy setting is enabled, the Universal Print Server components are installed, and the policy settings are applied.

<table>
<thead>
<tr>
<th>Policy setting</th>
<th>Interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client printer redirection, Auto-create client printers</td>
<td>After the Universal Print Server is enabled, client network printers are created using the Universal print driver instead of the native drivers. Users see the same printer name as before.</td>
</tr>
<tr>
<td>Session printers</td>
<td>When you use the Citrix Universal Print Server solution, Universal print driver policy settings are honored.</td>
</tr>
<tr>
<td>Direct connections</td>
<td>When the Universal Print Server is enabled and the Universal print driver usage policy setting is</td>
</tr>
</tbody>
</table>
To print server configured to use universal printing only, a direct network printer can be created to the print server, using the Universal print driver.

<table>
<thead>
<tr>
<th>Policy setting</th>
<th>Interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPD preference</td>
<td>Supports EMF and XPS drivers.</td>
</tr>
</tbody>
</table>

The Citrix Universal print driver used by the Universal Print Server disables the following user interface controls:
- In the Printer Properties dialog box, the Local Printer Settings button
- In the Document Properties dialog box, the Local Printer Settings and Preview on client buttons

When using the Universal Print Server, the Add Printer Wizard for the Citrix Print Provider is the same as the Add Printer Wizard for the Windows Print Provider, with the following exceptions:
- When adding a printer by name or address, you can provide an HTTP/SOAP port number for the print server. That port number becomes a part of the printer name and appears in displays.
- If the Citrix Universal print driver usage policy setting specifies that universal printing must be used, the Universal print driver name appears when selecting a printer. The Windows Print Provider cannot use the Universal print driver.

The Citrix Print Provider does not support client-side rendering.
Auto-create client printers

Apr 11, 2013
The autocreation feature creates a list of printers available to a user after logon.

Use the following Citrix policy settings to configure universal printing. For more information, refer to the on-screen policy settings help.

- **Universal print driver usage.** Specifies when to use universal printing.
- **Auto-create generic universal printer.** Enables or disables autocreation of the generic Citrix Universal Printer object for sessions when a user device compatible with Universal Printing is in use. By default, the generic Universal Printer object is not auto-created.
- **Universal driver preference.** Specifies the order in which the system attempts to use Universal print drivers, beginning with the first entry in the list. You can add, edit, or remove drivers and change the order of the drivers in the list.
- **Universal printing preview preference.** Specifies whether to use the print preview function for auto-created or generic universal printers.
- **Universal printing EMF processing mode.** Controls the method of processing the EMF spool file on the Windows user device. By default, EMF records are spooled directly to the printer. Spooling directly to the printer allows the spooler to process the records faster and uses fewer CPU resources.

For more policies, refer to [Optimize printing performance](https://docs.citrix.com).

To change the defaults for settings such as paper size, print quality, color, duplex, and the number of copies, refer to [How to Change the Default Settings on the Citrix Generic Universal Printer](https://docs.citrix.com).

At the start of a session, the system auto-creates all printers on the user device by default. You can control what, if any, types of printers are provisioned to users and prevent autocreation.

Use the Citrix policy setting **Auto-create client printers** to control autocreation. You can specify that:

- All printers visible to the user device, including network and locally attached printers, are created automatically at the start of each session (default)
- All local printers physically attached to the user device is created automatically
- Only the default printer for the user device is created automatically
- Autocreation is disabled for all client printers

The **Auto-create client printers** setting requires that the Client printer redirection setting is **Allowed** (the default).
Assign network printers to users

Mar 26, 2013

To create administrator-assigned printers, configure the Citrix policy setting Session printers. Add a network printer to that policy using one of the following methods:

- Enter the printer UNC path using the format `\servername\printername`.
- Browse to a printer location on the network.
- Browse for printers on a specific server. Enter the server name using the format `\servername` and click Browse.

Important: The server merges all enabled session printer settings for all applied policies, starting from the highest to lowest priorities. When a printer is configured in multiple policy objects, custom default settings are taken from only the highest priority policy object in which that printer is configured. Network printers created with the Session printers setting can vary according to where the session was initiated by filtering on objects such as subnets.

By default, the user's main printer is used as the default printer for the session. Use the Citrix policy setting Default printer to change how the default printer on the user device is established in a session.

1. On the Default printer settings page, select a setting for Choose client's default printer:
   - Network printer name. Printers added with the Session printers policy setting appear in this menu. Select the network printer to use as the default for this policy.
   - Do not adjust the user's default printer. Uses the current Terminal Services or Windows user profile setting for the default printer. For more information, refer to the on-screen policy settings help.
2. Apply the policy to the group of users (or other filtered objects) you want to affect.

Note: Proximity printing is also provided by the Citrix Universal Print Server and does not require the configuration described in this section.

1. Create a separate policy for each subnet (or to correspond with printer location).
2. In each policy, add the printers in that subnet’s geographic location to the Session printers setting.
3. Set the Default printer setting to Do not adjust the user's default printer.
4. Filter the policies by client IP address. Be sure to update these policies to reflect changes to the DHCP IP address ranges.
Manage printer drivers

Jun 18, 2013
To minimize administrative overhead and the potential for print driver issues, Citrix recommends use of the Citrix Universal print driver.
If auto-creation fails, by default, the system installs a Windows-native printer driver provided with Windows. If a driver is not available, the system falls back to the Universal print driver. For more information about printer driver defaults, refer to Default print operations.
If the Citrix Universal print driver is not an option for all scenarios, map printer drivers to minimize the amount of drivers installed on Server OS machines. In addition, mapping printer drivers enables you to:

- Allow specified printers to use only the Citrix Universal print driver
- Allow or prevent printers to be created with a specified driver
- Substitute good printer drivers for outdated or corrupted drivers
- Substitute a driver that is available on Windows server for a client driver name

The automatic installation of print drivers should be disabled to ensure consistency across Server OS machines. This can be achieved through Citrix policies, Microsoft policies, or both. To prevent the automatic installation of Windows-native printer drivers, disable the Citrix policy setting Automatic installation of in-box printer drivers.

Each client provides information about client-side printers during logon, including the printer driver name. During client printer autocreation, Windows server printer driver names are selected that correspond to the printer model names provided by the client. The autocreation process then uses the identified, available printer drivers to construct redirected client print queues.

The general process for defining driver substitution rules and editing print settings for mapped client printer drivers follows.

1. To specify driver substitution rules for auto-created client printers, configure the Citrix policy setting Printer driver mapping and compatibility by adding the client printer driver name and selecting the server driver that you want to substitute for the client printer driver from the Find printer driver menu. You can use wildcards in this setting. For example, to force all HP printers to use a specific driver, specify HP* in the policy setting.
2. To ban a printer driver, select the driver name and choose the Do not use setting.
3. As needed, edit an existing mapping, remove a mapping, or change the order of driver entries in the list.
4. To edit the printing settings for mapped client printer drivers, select the printer driver, click Settings, and specify settings such as print quality, orientation, and color. If you specify a printing option that the printer driver does not support, that option has no effect. This setting overrides retained printer settings the user set during a previous session.
5. Citrix recommends testing the behavior of the printers in detail after mapping drivers, since some printer functionality can be available only with a specific driver.

When users log on the system checks the client printer driver compatibility list before it sets up the client printers.
Optimize printing performance

Jun 18, 2013

To optimize printing performance, use the Universal Print Server and Universal print driver. The following policies control printing optimization and compression:

- Universal printing optimization defaults. Specifies default settings for the Universal Printer when it is created for a session:
  - Desired image quality specifies the default image compression limit applied to universal printing. By default, Standard Quality is enabled, meaning that users can only print images using standard or reduced quality compression.
  - Enable heavyweight compression enables or disables reducing bandwidth beyond the compression level set by Desired image quality, without losing image quality. By default, heavyweight compression is disabled.
  - Image and Font Caching settings specify whether or not to cache images and fonts that appear multiple times in the print stream, ensuring each unique image or font is sent to the printer only once. By default, embedded images and fonts are cached.
  - Allow non-administrators to modify these settings specifies whether or not users can change the default print optimization settings within a session. By default, users are not allowed to change the default print optimization settings.

- Universal printing image compression limit. Defines the maximum quality and the minimum compression level available for images printed with the Universal print driver. By default, the image compression limit is set to Best Quality (lossless compression).

- Universal printing print quality limit. Specifies the maximum dots per inch (dpi) available for generating printed output in the session. By default, no limit is specified.

By default, all print jobs destined for network printers route from the Server OS machine, across the network, and directly to the print server. Consider routing print jobs over the ICA connection if the network has substantial latency or limited bandwidth. To do that, disable the Citrix policy setting Direct connections to print servers. Data sent over the ICA connection is compressed, so less bandwidth is consumed as the data travels across the WAN.

While printing files from Server OS machines to user printers, other virtual channels (such as video) may experience decreased performance due to competition for bandwidth especially if users access servers through slower networks. To prevent such degradation, you can limit the bandwidth used by user printing. By limiting the data transmission rate for printing, you make more bandwidth available in the HDX data stream for transmission of video, keystrokes, and mouse data. Important: The printer bandwidth limit is always enforced, even when no other channels are in use.

Use the following Citrix policy Bandwidth printer settings to configure printing bandwidth session limits. To set the limits for the site, perform this task using Studio. To set the limits for individual servers, perform this task using the Group Policy Management Console in Windows locally on each Server OS machine.

- The Printer redirection bandwidth limit setting specifies the bandwidth available for printing in kilobits per second (kbps).
- The Printer redirection bandwidth limit percent setting limits the bandwidth available for printing to a percentage of the overall bandwidth available.
  Note: To specify bandwidth as a percentage using the Printer redirection bandwidth limit percent setting, enable the Overall session bandwidth limit as well.
  If you enter values for both settings, the most restrictive setting (the lower value) is applied.

To obtain real-time information about printing bandwidth, use Citrix Director.

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.286
The following table summarizes where you can display printers and manage print queues in your environment.

<table>
<thead>
<tr>
<th>Client printers (Printers attached to the user device)</th>
<th>Printing Pathway</th>
<th>UAC Enabled?</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client printing pathway</td>
<td>On</td>
<td>Print Management snap-in located in the Microsoft Management Console</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Off</td>
<td>Pre-Windows 8: Control Panel Windows 8: Print Management snap-in</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network printers (Printers on a network print server)</th>
<th>Printing Pathway</th>
<th>UAC Enabled?</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network printing pathway</td>
<td>On</td>
<td>Print Server &gt; Print Management snap-in located in the Microsoft Management Console</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Off</td>
<td>Print Server &gt; Control Panel</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network printers (Printers on a network print server)</th>
<th>Printing Pathway</th>
<th>UAC Enabled?</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client printing pathway</td>
<td>On</td>
<td>Print Server &gt; Print Management snap-in located in the Microsoft Management Console</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Off</td>
<td>Pre-Windows 8: Control Panel Windows 8: Print Management snap-in</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Local network server printers (Printers from a network print server that are added to a Server OS machine)</th>
<th>Printing Pathway</th>
<th>UAC Enabled?</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network printing pathway</td>
<td>On</td>
<td>Print Server &gt; Control Panel</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Off</td>
<td>Print Server &gt; Control Panel</td>
<td></td>
</tr>
</tbody>
</table>

Note: Print queues for network printers that use the network printing pathway are private and cannot be managed through the system.
Manage

Mar 12, 2015
This section supports and provides information about the managing features, including:

- **Provisioning virtual desktops**  
  Creating and managing machine catalogs

- **Allocating desktops**  
  Creating desktop groups, managing catalogs and desktop groups, importing and exporting user data, and other general management

- **Managing controller environments**  
  Controller Discovery, adding, removing, and relocating controllers, configuring SSL

- **Configuring hosts**  
  Creating, editing and deleting hosts, editing and deleting connections, managing machines, and enabling maintenance mode
Machines

Jun 20, 2013

Use machine catalogs to efficiently manage machines as follows:

- Update users' desktops
- Add machines available for users
- Modify a machine catalog
- Delete a machine catalog
- Manage computer accounts:
  - Active Directory accounts for Desktop OS and Server OS machines
  - Organizational Units (OU) for Remote PC Access machines
- Manage machine catalogs whose machines have older operating systems or Virtual Desktop Agent (VDA) versions including:
  - Windows XP or Windows Vista operating systems
  - VDAs whose versions are earlier than this release

Random machine catalogs

Maintain users' desktops by applying global updates, such as Windows updates, anti-virus software updates, or configuration changes, to the Master Image. Then modify the machine catalog to use the updated master image so users receive the updated desktop at their next logon. Using this method lets you make significant changes to users' desktops, including upgrading to a new operating system, for large numbers of users in one operation.

Static and Remote PC Access machine catalogs

You must manage updates to users' desktops outside of Studio, either on an individual basis or collectively using third-party software distribution tools. For machines created through Provisioning Services, updates to users' desktops are propagated through the vDisk.

Citrix recommends that you save copies or snapshots of master images before you make any updates. The Database retains a historical record of the master images used with each machine catalog. To quickly revert a machine catalog to use the previous version, preserving the old Master Images in their original state, that is, do not delete, move, or rename the master images. You can then use the previous version of the Master Image should users encounter problems with updates that you have deployed to their desktops, thereby minimizing user downtime.

You can deploy additional machines to an existing machine catalog as follows:

<table>
<thead>
<tr>
<th>For this type of machine catalog:</th>
<th>You can:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desktop OS and Server OS</td>
<td>Add more machines and, if required, add more Active Directory computer accounts.</td>
</tr>
</tbody>
</table>
Remote PC Access

For this type of machine catalog:

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deploy the VDA component on additional office PCs that already are members of appropriate Organizational Units (OUs).</td>
<td></td>
</tr>
<tr>
<td>Existing and physical</td>
<td>Set up additional VMs or blade PCs, respectively, plus any computer accounts that are required, outside of Studio. You can then add these machines and accounts to the machine catalog.</td>
</tr>
<tr>
<td>Machines created by Provisioning Services (with Personal vDisks)</td>
<td>Add more machines by joining more target devices to an existing device collection using Provisioning Services. Alternatively, create additional device collections in Provisioning Services and then add the new collections to the existing machine catalog.</td>
</tr>
</tbody>
</table>

- **Modify a machine catalog** — You can add or remove administrators from the list of assignment administrators permitted to use the machine catalog, edit the machine catalog name or description, and quickly view the details and status of all the machines included in the machine catalog.
- **Delete a machine catalog** — When you delete a machine catalog, the machines and the associated Active Directory computer accounts or OUs are removed from Studio management. For Windows Desktop OS and Windows Server OS machine catalogs, you can optionally delete the machines and computer accounts from the host and from Active Directory.
- **Upgrade a machine catalog** — If a machine catalog includes any machines running Windows XP or Vista, or includes VDA versions prior to this release, the wizard detects and notifies you that these machines reside in a separate machine catalog. You can continue to use these early version machine catalogs and their Delivery Groups. However, to take advantage of new features introduced in this release, you must upgrade as described in **Upgrade a machine catalog** and as follows:
  - Upgrade the machine's operating system to Windows 7 or Windows 8, which is compatible with XenDesktop 7
  - Upgrade any of the following VDA versions to XenDesktop 7:
    - 5.0 Service Pack 1
    - 5.5
    - 5.6
    - 5.6 Feature Pack 1
Add machines to a machine catalog

Jun 20, 2013

Before you add machines to a machine catalog, consider the following:

- Make sure that the virtualization infrastructure hosting the specified master image for the machine catalog has sufficient processors, memory, and storage to accommodate the additional machines you plan to create.
- Make sure you have enough computer accounts
  - Sufficient number of unused Active Directory computer accounts for the additional machines you plan to create. If you are using existing computer accounts, note that the number of machines you can create is limited by the number of accounts that are available.
  - Access to an Active Directory domain administrator account for the domain of which the desktops are members.
  - Adequate numbers of Organizational Units (OUs) for Remote PC Access machines.
- Remote PC Access machine catalogs — When Remote PC Access is first deployed, the system creates a machine catalog and associated Delivery Group specific for Remote PC Access. The machine catalog contains all of the Machine Accounts, which are individual accounts or Organizational Units (OUs), that contain multiple accounts. The Delivery Group contains all of the users assigned during the Remote PC Access deployment. In addition to adding machines to the catalog, you can add, change, or remove OUs to specify which additional machines are automatically added to the deployment from this point on.

1. Log on to the computer running Studio. If you plan to use Studio to create Active Directory computer accounts for the additional machines, log on using a domain administrator account for the domain of which the desktops are members.
2. Start Studio.
3. Select Machine Catalogs and then select a machine catalog in the results pane, and click Add machines.
4. On the Machines and Users page, import machines using one of the following methods:
   - Click Add Computers to select computers from the domain.
   - Select an Active Directory Account from the list.
   - If more Active Directory computer accounts are required and you want Studio to create new accounts for the machines, select Import accounts.
5. On the Import accounts page, provide the required information.
   To create new computer accounts, specify:
   - Active Directory to which the accounts are added
   - Domain
   - OU
   - Name for the new account
   To use existing accounts, click Browse and select computer accounts in Active Directory or click Import and specify a .csv file containing a list of account names. Because Studio manages these accounts, either allow Studio to reset the passwords for all the accounts or supply the account password (which must be the same for all accounts). Make sure that you import enough accounts for all additional machines you want to create.
6. On the Summary page, check that the details are correct.
   Studio performs machine creation as a background process so that you can continue to work. Because VMs are created sequentially, this can be a lengthy process when you add a large number of machines to a machine catalog. Machine
creation continues to completion even if you close Studio.
Modify a machine catalog

Mar 29, 2013

You can perform the following machine catalog changes with Edit Catalog:

- Change the machine catalog description.
- Change the machine catalog name.
- For Remote PC Access machine catalogs you can also:
  - Add, change, or remove Organizational Units (OUs), to modify access for the machines associated with the OU.
  - Change the machines associated with the Delivery Group, as described in Edit a Delivery Group.

1. In Studio, in the Machine Catalogs node, select the machine catalog you want to change.
2. Click Edit Machine Catalog. If this is not a Remote PC Access machine catalog, go to step 4.
3. For a Remote PC Access machine catalog, click Organizational Units. Add or remove Organizational Units, which contain multiple user accounts, and then click Save.
4. Click Description to make changes in descriptions that users and administrators see.

1. In Studio, in the Machine Catalogs node, select the machine catalog you want to change and then click Rename Machine Catalog.
2. Enter the name you want and then click OK.

To search for machine catalogs, use the Search node as described in Find machines, sessions, machine catalogs, applications, and Delivery Groups. You cannot search within the machine catalogs node.
Delete a machine catalog

Jun 20, 2013

Any administrator who has permissions to use a machine catalog when allocating desktops to users can delete that machine catalog. Before deleting a machine catalog, make sure that:

- All users are logged off from the desktops in the machine catalog (You can force log offs if users have not logged off )
- No disconnected user sessions are still running
- All machine catalogs machines are in maintenance mode
- All existing machines are shut down
- The machine catalog is not associated with a Delivery Group

1. In Studio, select the Machine Catalogs node and then select the machine catalog in the results pane, and click Delete Machine Catalog.

2. For Desktop OS and Server OS machine catalogs, specify whether to delete the machines hosting users' desktops or applications. If you delete the machines, you have the following options for the removed machines and the associated Active Directory computer accounts:
   - Leave the machines in the Active Directory
   - Disable the machines in the Active Directory
   - Delete the machines from the Active Directory

3. On the Summary page, check that the details are correct.
Delete machines from machine catalogs

Jun 20, 2013

When you delete a machine, users no longer can access it, and the machine is deleted from the machine catalog. Before deleting a machine, make sure that all user data is backed up or no longer required. You can delete a machine only when no users are logged on. To temporarily stop users from connecting to a machine while you are deleting it, put the machine into maintenance mode.

1. In Studio, use Search to locate the machine you want to delete, or select a Delivery Group and click View Machines.  
2. Select the machine, and put it in maintenance mode.  
3. Click Delete and select one of the following options:  
   - Remove the virtual machines from the machine catalog without deleting the virtual machines.  
   - Remove the virtual machines from the machine catalog and deleting the virtual machines. You must also select how to handle the associated Active Directory accounts:  
     - Leave the accounts in the machine catalog and do not change them in Active Directory.  
     - Remove the accounts from the machine catalog but do not remove them from Active Directory.  
     - Remove the accounts from the machine catalog and disable them in Active Directory.  
     - Remove the accounts from the machine catalog and delete them from Active Directory.
Manage desktop computer accounts

Sep 02, 2014
When managing computer accounts for desktops, you can:

- Free unused accounts to use for other desktops by removing Active Directory computer accounts from Desktop OS and Server OS machine catalogs, and Organizational Units (OUs) from Remote PC Access machine catalogs.
- Attach additional accounts to a machine catalog so that when more machines are added to this machine catalog, the computer accounts are already in place.

Follow these steps to manage Active Directory user accounts for Desktop OS and Server OS machine catalogs.

1. In Studio, select the Machine Catalogs node.
2. Select a machine catalog in the results pane and click Manage AD accounts.
3. On the Active Directory computer accounts page, select one of the following:
   - Create new Active Directory account:
     - Select the Active directory location by selecting a domain and folder.
     - Select an account name.
   - Use existing Active Directory accounts:
     - Browse to an account, or import an account.
   - Manage the account passwords:
     - Reset all account passwords - use this if you do not know the current passwords for the account; you must have permission to perform a password reset.
     - Enter a password that applies to all accounts - this will change the password on the accounts as they are imported.

Follow these steps to manage Organizational Units (OUs) and user machine accounts for Remote PC Access machine catalogs.

1. In Studio, select the Machine Catalogs node.
2. Select a Remote PC Access machine catalog in the results pane, click Edit Machine Catalog, and then select Users.
3. On the Select machine accounts for your users page, select or add user accounts:
   - Select an existing machine account or OU.
   - Add machine accounts or OUs.
Update a master image

May 07, 2013

To apply changes (for example, implement software updates) to all the desktops and applications from a machine catalog, update the master image. Managing the common aspects of users' desktops through a single master image lets you deploy system-wide changes, such as applying Windows updates or making configuration changes, to a large number of desktops very quickly. This lets you provide users with continuous access to their desktops so that they can continue working while any problems with the update are addressed.

Cloud deployments use templates in place of master images. Refer to the following information for preparing templates:

- For Amazon Web Services (AWS), see the
  — XenApp or XenDesktop Infrastructure Stack Creation using the CloudFormation template section in Deploy XenApp and XenDesktop 7.5 with Amazon VPC.
- For Citrix CloudPlatform, see

If the master image includes the latest VDA version, it enables the new VDA features.

Once you have prepared and tested a new or updated master image, modify the machine catalog to use the new master image. Desktops are updated with the new master image the next time users log off.

Note: After updating the master image, you must restart the machines through Studio for the changes to take effect and be available to your users. This may occur automatically; for example, when a user logs out of a random Desktop OS machine, or it may occur as part of a configured reboot schedule. Alternatively, you can restart a machine by finding and selecting it through Studio's Search node.

Before you update the master image, create a snapshot. Doing so lets you quickly revert a machine catalog to use the previous version of the master image should users encounter problems with updates.

Although Studio can create a snapshot, Citrix recommends that you create a snapshot using the Hypervisor Management console, and then select that snapshot in Studio. Using this method lets you provide a meaningful name and description rather than an automatically generated name.

Note the following

- For GPU master images, you can change the master image only through XenServer's XenCenter console. See Prepare a master image for information about GPU master images.
- For Provisioning Services (PVS) machine catalogs, you must publish a new vDisk to apply changes to the Provisioning Services machine catalogs. For detailed information, see Updating vDisks and Versioned vDisk Upgrade.

1. In Studio, select the Machine Catalogs node, select a machine catalog in the results pane, and click Update Machine.
2. On the Master Image page, select the host and the new or updated master image that you want to use.
   - For XenDesktop 7.x machine catalogs, a message prompts you to select a snapshot or virtual machine built using a 7.x VDA.
   - For machine catalogs running Windows XP or Windows Vista, an earlier VDA version is installed on Windows XP or Windows Vista machines.
3. On the Rollout Strategy page, specify how the new or updated master image is applied to users' desktops and click Next.

<table>
<thead>
<tr>
<th>If you are deploying:</th>
<th>Select:</th>
</tr>
</thead>
<tbody>
<tr>
<td>A non-urgent update and you want to minimize disruption to users</td>
<td>On the next restart (not right now) to apply the update when users next log off.</td>
</tr>
<tr>
<td>A non-urgent update and you want to inform users</td>
<td>Notify users of the update and enter a message. Users see the specified message and the update is applied when they next log off.</td>
</tr>
<tr>
<td>A critical update and you want to apply it to all users' desktops urgently</td>
<td>Immediately (restart the machine now) to automatically log off users and restart their desktops. Select Distribution time in which you can restart all the machines at once or restart over designated time intervals such as 30 minutes, one hour, or two hours.</td>
</tr>
<tr>
<td>A critical update and you want to apply it to all users' desktops urgently and you want to inform users</td>
<td>Notify users of the update, and specify the time delay before applying the update. The timer starts when Studio finishes making a temporary copy of the new or updated master image in the appropriate location Enter a message that users see when they next log on, or if the specified time limit is reached, users are automatically logged off and their desktops restarted.</td>
</tr>
</tbody>
</table>

4. On the Summary page, check that the details are correct and click Finish.

A historical record of the master images for each machine catalog is stored in the Database. This lets you revert a machine catalog and use the previous version of the master image should users encounter problems with updates that you have made. Desktops revert to the previous master image the next time users log off.

If you delete, move, or rename any previous master images, including any snapshots in the chains leading to the master images, you cannot revert the machine catalog to use them. When Studio is unable to locate the previous master image, you can browse for an alternative master image from which to update the desktops.

1. In the Studio, select the Machine Catalogs node, select your machine catalog in the results pane, and click Rollback machine update.

2. On the Strategy page, specify how to apply the reverted master image to users' desktops.

<table>
<thead>
<tr>
<th>If reverting:</th>
<th>Select:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users' desktops is not urgent and you want to minimize disruption</td>
<td>None. Users' desktops are reverted when they next log off.</td>
</tr>
<tr>
<td>To users</td>
<td>Select:</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
</tr>
<tr>
<td>Users' desktops is not urgent and you want to inform users</td>
<td>Send message and enter a message. Users see the specified message and their desktops are reverted when they next log off.</td>
</tr>
<tr>
<td>Users' desktops is critical and you want to revert all users' desktops urgently,</td>
<td>Restart immediately to automatically log off users and restart their desktops.</td>
</tr>
<tr>
<td>Users' desktops is urgent and you want to allow users some time to save their work before reverting their desktops</td>
<td>Send message then restart after delay. Enter a message and specify the time delay before reverting the desktops. The timer starts when Studio finishes making a temporary copy of the reverted master image in the appropriate location. Users see the specified message and their desktops are reverted when they next log off or, if the specified time limit is reached, users are automatically logged off and their desktops restarted.</td>
</tr>
</tbody>
</table>

The rollback strategy is only applied to desktops that need to be reverted. Users of desktops that have not been updated with the problematic master image that prompted the rollback, for example because the user has not logged off, do not receive any messages and are not forced to log off.

3. On the Summary page, check that the details are correct and then click Finish.
Upgrade a machine catalog

Apr 22, 2014

Machines in a machine catalog from previous versions might contain machines running earlier VDA versions, or earlier Windows operating systems. These include:

- VDA versions 5.0 Service Pack 1, 5.5, 5.6, or 5.6 Feature Pack 1
- VDA Version 5.6 on a Microsoft Windows 7 operating system
- Microsoft Windows XP or Microsoft Windows Vista operating systems

Note: Remote PC Access does not support Microsoft Windows Vista.

Perform the following upgrades on machines before you upgrade the machine catalog:

- Upgrade the operating system to Microsoft Windows 7 or Microsoft Windows 8.x.
- Upgrade the VDA version to version 7 as described in XenDesktop 7 upgrade components and XenDesktop 5 upgrade factors.

Once you upgrade these machines' operating systems or VDA versions, upgrade their machine catalogs to take advantage of the latest features in this release.

After upgrading the machine operating system and VDA version, and before you run Studio to upgrade the machine catalog, you must:

- Make sure you upgrade the VDA version in the Provisioning Services console if you use Provisioning Services to create machine catalogs.
- Start the machines and register them with the Delivery Controller. Otherwise, the Studio wizard cannot determine whether the machines in the machine catalog need upgrading.

When creating or editing a machine catalog, the wizard displays the following option if it detects older operating system versions:

Some virtual machines are running Windows XP or Windows Vista.

You should also select this option if the virtual machine is running Windows 7 with VDA 5.6

Select the checkbox for this option so that the wizard creates a machine catalog specifically for these types of systems.

You can create the machine catalog, but before upgrading the machine catalog, make sure that referenced machines have been upgraded to:

- Windows 7 or Windows 8.x
- For machines running Windows 7, they must also run VDA version 7.

1. In Studio, select the Machine Catalogs node and then click View.
2. Select the machine catalog to upgrade. Check the Details tab, which displays operating system and VDA version information and click Upgrade Catalog. The wizard checks to make sure that the machine catalog can be upgraded and indicates that the machine catalog is suitable for upgrade.
   - If the wizard indicates that the machine catalog is suitable for upgrade, an informational message appears. Click
Upgrade and follow the prompts to complete the upgrade.

- If one or more machines are not suitable, the wizard lists the reasons for the machine deficiencies.
- Click View Details for detailed information. You can optionally export the information to a file by clicking Export Details in the Upgrade Warnings dialog box.
- Click Cancel and update the machines that need upgrades.

Citrix recommends that you resolve machine issues before upgrading the machine catalogs to ensure that all machine function properly.

You can revert the machines in a machine catalog to their prior states.

If you used Provisioning Services to create the machine catalog you want to revert, make sure you change the VDA version in the Provisioning Services console before reverting the machine catalog upgrade.

1. In the Machine Catalogs node, click View.
2. Select the machine catalog to revert. Check the Details tab, which displays operating system and VDA version information and click Undo. Follow the prompts to complete the process.
Once you have created Delivery Groups to deliver your applications and desktops, you can perform the following Delivery Group management tasks:

- Edit a Delivery Group
- Create a Delivery Group application
- Change the display properties of Delivery Groups
- Power manage Desktop OS machines
- Import and export user data
- Manage machine sessions
- Stop users from connecting to machines
- Manage Delivery Group resources
- Delete machines from machine catalogs
- Remove machines from Delivery Groups
- Find machines, sessions, machine catalogs, and Delivery Groups
- Secure Delivery Groups
- Restrict access to machines in a Delivery Group
- Shut down and restart machines
- Remove the Shut Down command
- Test a Delivery Group
- Upgrade a Delivery Group
- Set up the restart schedule for Server OS Machine desktops
- Manage Server OS Machine server load
Edit a Delivery Group

You can change a Delivery Group's characteristics.

Note: When editing a Remote PC Access Delivery Group, the User Profile Selection and Reboot Schedule do not apply and are locked.
In Studio, select the Delivery Group node, select a Delivery Group, and click Edit Delivery Group. The following table summarizes Delivery Group changes.

<table>
<thead>
<tr>
<th>On this page:</th>
<th>Change:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users</td>
<td>Users or user groups that can access desktops and applications. Select user groups by browsing or entering a list of Active Directory users and groups each separated by a semicolon. For Desktop OS Delivery Groups, import user data from a file after you create the group.</td>
</tr>
<tr>
<td>User Assignments</td>
<td>Machines assigned to the Delivery Group by selecting and assigning machines. You can optionally assign users as described in the Users row.</td>
</tr>
<tr>
<td>Delivery Type</td>
<td>Depending on the catalog type, select what the machines deliver to users:   - Desktops only.   - Applications only.   - Desktops and applications. This option is not available in static Desktop OS machines.</td>
</tr>
<tr>
<td>StoreFront</td>
<td>Select StoreFront URLs to be pushed to Citrix Receiver so that Receiver can connect to a store without user intervention. This feature is not available for Application-only Delivery Groups.</td>
</tr>
<tr>
<td>Scopes</td>
<td>Select or change the Delivery Group's scope. Scopes are groups of objects that represent parts of your deployment and your organization. For example, you might have a scope for Delivery Groups used by a Sales team, and another that you assign to the Support team. Make sure that you select the correct scopes so that the right users can access the desktops in this Delivery Group.</td>
</tr>
<tr>
<td>End User Settings</td>
<td>Change the Delivery Group description, number of desktops per user, color depth, time zone, or Secure ICA setting.</td>
</tr>
<tr>
<td>Power</td>
<td>Manage the following machine power settings:</td>
</tr>
</tbody>
</table>
Management

On this page:

<table>
<thead>
<tr>
<th>Change:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Days and times to turn off machines.</td>
</tr>
<tr>
<td>Specify the delay (in minutes) before suspending any disconnected machine for peak hours and off peak hours.</td>
</tr>
<tr>
<td>Specify what happens to the machines when disconnected: nothing, suspend, or shut down.</td>
</tr>
</tbody>
</table>

Access Policy

Change whether the Delivery Group connections go through the NetScaler Gateway.

Specify which machines connect through the NetScaler Gateway using filters.

Restart Schedule

Change the following settings for Server OS Delivery Groups:

- Whether and how to communicate the reboot to users.
- Automatically reboot machines, and the schedule to do so if selected.

When first created, Remote PC Access machine catalogs have an association to a Delivery Group. This means that machine accounts or Organizational Units added to the machine catalog later can be added to the Delivery Group. The association can be switched off or on.

1. In Studio, select the Delivery Group node in the left pane, select a Remote PC Access Delivery Group.
2. In the Details section, select Machine Catalogs.
3. Select a machine catalog and then click Remove Association to remove the machine catalog association from the Delivery Group.
4. You can add or restore an association by clicking Add Desktops.

To search for Delivery Groups, use the Search node as described in Find machines, sessions, machine catalogs, applications, and Delivery Groups. You cannot search within the Delivery Group tab.
Create a Delivery Group application

Mar 28, 2015

Using the Studio Create Application feature, you can select applications for a Delivery Group that your users can access when they log on to the desktop. Studio discovers the applications on the master images or templates. You can also manually add custom applications, which reside on Delivery Group machines. The following application types are available:

- **Applications installed on Delivery Group machines** — The Studio wizard discovers these applications and then makes them available to the Delivery Group’s desktops. The following types of applications may be discovered and made available:
  - 16-bit and 32-bit applications hosted on machines running Windows XP or Windows 7
  - Applications hosted on App-V servers
  - Applications installed on users’ machines
- **Custom applications:**
  - Any applications that were not discovered by the wizard that you want to include.
  - Applications installed on the user device. Once selected, these applications appear on the desktop’s Windows Start menu.
- **Off-line App-V applications** — If you plan to be disconnected from the network for an extended period of time, you can work in offline mode through the Microsoft App-V client. Follow the instructions described in How to Work Offline or Online with Application Virtualization.
  
  Once the App-V 5 client is installed on the end point from which you want to offer offline App-V applications, the applications published to the user through the App-V 5 Management Server become available to that user.

Note: You cannot create applications for Remote PC Access Delivery Groups.

1. In Studio, select the Delivery Group node, select the Applications tab, and then click Create Applications.
2. On the Delivery Group page, select a Delivery Group to host the applications and then click Next.
3. On the Applications page, add existing applications to this Delivery Group from the list of applications Studio discovers and displays.
   
   Note: The assigned user or group must be a member of the Delivery Group to which this application is assigned to view or access this application.
4. To optionally add a custom application, click Add applications manually then click Next. Complete the following information about the application on the Add an Application Manually page:
   - Path to the executable file
   - Optional command line parameters
   - Directory
   - Display names for administrator and users — Name that users see and Name that only administrators see
   - Location — Path to the application's executable file, working directory, and an optional command line argument
   - Limit Visibility — Limit which users or groups of users can see the application
     
     Note: The assigned user or group must be a member of the Delivery Group to which this application is assigned. Otherwise, they cannot view or access this application.
   - File Type Association — Which types of files the application automatically opens
5. On the Summary page, check all details
6. Repeat the previous steps to create applications for additional application Delivery Groups or application and desktop Delivery Groups.
Modify and manage applications

Dec 30, 2013

To modify and manage applications:

1. Select the Delivery Groups node, then select the Applications tab.
2. Select the application on the Applications tab.
3. Choose a task on the Actions pane.

Application modifications might not take effect for users connected to the application until the users have logged off their sessions.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refresh the applications listing</td>
<td>Select Refresh in the Applications tab.</td>
</tr>
<tr>
<td>View or change application settings</td>
<td>Select the application you want to edit, then click Properties. Use the Application Settings wizard to change the application's settings. For information about configuring settings, see <a href="https://docs.citrix.com">Create a Delivery Group application</a>.</td>
</tr>
<tr>
<td>Copy</td>
<td>Select the application you want to copy, and then click Copy Application. A copy of the application appears in the Applications tab.</td>
</tr>
<tr>
<td>Disable</td>
<td>Make sure that the application is not in use. Select the application on the Applications tab and then click Disable from the Action pane. The application appears as Disabled in the Applications tab.</td>
</tr>
<tr>
<td>Rename</td>
<td>Select the application on the Applications tab and then click Rename from the Action pane. Enter the new name in the dialog that appears.</td>
</tr>
<tr>
<td>Add or edit searchable tags assigned to and used to categorize applications</td>
<td>Select the application on the Applications tab then click Edit Tags. Add or edit tags in the Edit Applications Tags dialog that appears.</td>
</tr>
<tr>
<td>Delete applications associated with a Delivery Group</td>
<td>Make sure that the application is not in use. Select the application on the Applications tab then click Delete.</td>
</tr>
<tr>
<td><strong>Action</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Search</td>
<td>On the Applications tab, select a saved search from the Saved searches drop-down menu or create a new search. For information about searching, see <a href="https://docs.citrix.com">Find machines, sessions, machine catalogs, applications, and Delivery Groups</a>.</td>
</tr>
</tbody>
</table>

Deleting the application removes the application assignment from the Delivery Group, it does not remove it from the master image.
Change Delivery Group user settings

Mar 29, 2013
You can change the following user settings for all the machines in a Delivery Group.

1. In Studio, select the Delivery Groups node and select the Delivery Group whose properties you want to change.
2. Click Edit Delivery Group.
3. Click End user settings and, change the desktops' properties as required.

You can change the following user properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Delivery Group description that users see and in StoreFront.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Makes the Delivery Group available to provide resources.</td>
</tr>
<tr>
<td>Desktops per user (Desktop OS machines only)</td>
<td>Limits the number of shared desktop instances that a user can simultaneously launch. For the assigning-on-first-use case, this property limits the number of desktops users can assign to themselves.</td>
</tr>
<tr>
<td>Color depth</td>
<td>Note that the graphics driver on the Virtual Desktop Agent handles Alpha (transparency) data in addition to red, green, and blue data. Assuming that a suitable Citrix plug-in client in use has enough graphics memory to display 32-bit color, sessions are displayed at that color depth even if you select True Color (24 bit) here.</td>
</tr>
<tr>
<td>Time zone</td>
<td>The time zone in which the Delivery Group operates.</td>
</tr>
<tr>
<td>Enable Secure ICA</td>
<td>Conceals all communications to and from machines in the Delivery Group using the SecureICA feature, which encrypts the ICA protocol.</td>
</tr>
</tbody>
</table>
Power manage Desktop OS machines

Aug 29, 2013

Full and partial machine power management is available for Delivery Group machines.

Note the following:

- This procedure only applies to Desktop OS machines. For Server OS, manage power settings by using reboot schedules. See [Set up the restart schedule for a Server OS machine Delivery Group](https://docs.citrix.com) for detailed information. You cannot power manage remote PC access Delivery Group machines.

- You can only power manage virtual machines, not physical ones. The ability to fully or partially control power management depends on how the Delivery Group's virtual machines are allocated to users or user devices. Permanently allocated machines can only be partially power managed. Machines using GPU capabilities cannot be suspended. Therefore, attempting to power off a Delivery Group with GPU machines results in an error.

Machines can be in one of these states:

- In random Delivery Groups: randomly allocated and in use
- In static or random Delivery Groups: unallocated (and therefore unconnected)
- In static Delivery Groups:
  - Permanently allocated and unconnected (but ready to be connected)
  - Permanently allocated and in use

At any time, static Delivery Groups typically contain both permanently allocated and unallocated machines. Initially, all the machines are unallocated (apart from any manually allocated to individuals when the Delivery Group was created). As users connect, some get permanently allocated. Therefore, when you fully power manage groups of this type, you are only fully managing the unallocated machines in it. The permanently allocated machines are partially managed.

**Pools and buffers**

For random Delivery Groups and unallocated machines in static Delivery Groups, a pool is a set of unallocated (or temporarily allocated) machines in the Delivery Group that are kept in a powered-on state, ready for users to connect. When a user logs on, they are immediately presented with a machine. The pool size (the number of machines kept powered on) is configurable. For example, you set up a bigger pool during office hours. For static Delivery Groups, there is no pool in Studio but you can use the XenDesktop SDK to configure one.

A buffer is an extra, standby set of unallocated machines that are turned on, ready for users to connect. For random Delivery Groups and unallocated machines in static Delivery Groups, machines in the buffer are turned on when the number of machines in the pool drops below the threshold set by the buffer size. This is a percentage of the Delivery Group size (by default, 10%). For large Delivery Groups, a significant number of machines may be turned on when the threshold is exceeded, so plan your Delivery Group sizes accordingly or adjust the default buffer size using the SDK.

**Power state timers**

You can suspend machines after users have disconnected for a defined time using power state timers. For example, machines can suspend automatically outside office hours if users have been disconnected for at least 10 minutes. Unless you have configured the ShutdownDesktopsAfterUse property of a Delivery Group using the SDK, random machines or
machines with Personal vDisks are always automatically shut down when users log off.

You can configure the timers separately for weekdays (by default, Monday to Friday) and weekends, and for peak and off-peak periods. The peak period covers the time at which most users log on to their desktops, and starts at the beginning of a business day. Use the SDK if you want to shut down, rather than suspend, machines in response to power state timers, or if you want the timers to be based on logoffs, rather than disconnections. Also, note that the Weekdays and Weekend selections in this procedure are defaults that can be configured using the SDK. For information about using the XenDesktop SDK, see About the XenDesktop SDK.

Partial power management of permanently allocated machines

With machines permanently allocated to individuals or user devices, you can set power state timers but not pools or buffers. XenDesktop turns on the machines at the start of each peak period, and turns them off at the start of each off-peak period, so you have no fine control (as you do with unallocated machines) over the number of machines that become available to compensate for machines that are consumed.

To power manage Delivery Groups

1. In Studio, select the Delivery Group node, and then select the Delivery Group whose power management settings you want to control.
2. Click Edit Delivery Group.
3. Click Power management and then select Weekdays in Power on/off machines.
4. For random Delivery Groups, in Machines to be powered on, click Edit and specify the pool size during weekdays.
5. In Machines to be powered on, select the number of machines to power on.
6. In Peak hours, set your organization's peak and off-peak hours during weekdays.
7. Set power state timers for peak and non-peak hours during weekdays:
   - In During peak hours When disconnected, specify the delay (in minutes) before suspending any disconnected machine in the Delivery Group, and select Suspend.
   - In During off-peak hours When disconnected, specify the delay before turning off any logged-off machine in the Delivery Group, and select Shutdown. This timer is not available for groups based on random machines.
8. Select Weekend.
9. Configure, as previously described, the peak hours, and power state timers for weekends.
Import and export user data

Feb 20, 2013

You can allocate desktops and applications to users by importing data from a .csv file if:

- The Delivery Group is based on existing or physical machines
- You have correct permissions to access the file and the Delivery Group

This .csv file can contain data from any previous version, and you can only use it to update Delivery Groups based on physical machines.

You can also export user data to a file.

Import and export files must have the following characteristics:

- They must be .csv files.
- The first line in the file must contain column headings, which can be:
  - [ADComputerAccount],[AssignedUser],[VirtualMachine],[HostId].
  - The column headings can be in any order, but they must be comma-separated.
- The subsequent lines contain the appropriate data, also comma-separated. The ADComputerAccount entries can be any of the following:
  - Common names (for example computer01)
  - IP addresses (for example 10.50.10.80)
  - Distinguished names (for example computer01.mydomain.com)
  - Domain and computer name pairs (for example mydomain\computer01)

You can find sample files on the installation media in \support\ImportExport.

To import data from or export data to a file

1. In Studio, under Delivery Groups, select the Delivery Group whose data you want to import or export.
2. Click Edit Delivery Group.
3. On the Edit Assignment page, click Import list or Export list.
Manage machine sessions

Mar 29, 2013

When a user logs on to a machine, the user device links to the Virtual Delivery Agent (VDA on the machine and establishes a session. When performing maintenance or to assist users, you can control sessions by:

- Logging users off sessions
- Disconnecting sessions
- Sending messages to users
- Searching to locate sessions, users, and machines

To log off or disconnect sessions

Depending on the machine type, you can log off and disconnect sessions.

- If you log off a session, the session closes and the machine becomes available to other users unless it is allocated to a specific user.
- If you disconnect a session, the user's applications continue to run and the machine remains allocated to that user. If the user reconnects, the same machine is allocated.

Note: Depending on the session's machine type, you can configure power state timers to automatically process unused sessions. This frees up machines and saves power. For example, you can set up the system to automatically log off any disconnected session after 10 minutes.

1. In Studio, use Search to locate the session or select a Desktop Group and click View Machines.
2. Select the session or desktop and click Log off or Disconnect.

To send messages to users

You can send messages to users to inform them about machine maintenance that affects their sessions. For example, you may want to tell users to log off before critical maintenance is about to occur.

1. In Studio, use Search to locate the session, machine, or user. Alternatively, select a Desktop Group and click View Machines.
2. Select the session, machine, or user and click Send message.
3. Compose the message.
Enable or disable maintenance mode

Jun 20, 2013
To temporarily stop connections to a machine to perform maintenance tasks, put the machine into maintenance mode. You can put Delivery Groups or individual machines into maintenance mode.

Putting a machine into maintenance mode lets you perform administrative tasks on the associated image, such as applying patches and upgrades using image management tools.

User connectivity is affected as follows when in maintenance mode:

- With Server OS machines, users can connect to existing sessions but cannot start new sessions.
- With Desktop OS and Remote PC Access machines, users cannot connect or reconnect once the machine is in maintenance mode. If they are already connected, then they stay connected until they next disconnect or log off.

Machines are available for user connections when you take them out of maintenance mode.

1. In Studio, use Search, or select a Delivery Group and click View Machines to locate individual machines. Alternatively, select the Delivery Groups node to view and locate a Delivery Group.
2. Select the machine or Delivery Group and click Enable maintenance mode or Disable maintenance mode.
Manage Delivery Group resources

Mar 21, 2015

Machines might contain personal data that you must manage before reassigning the machines. For example, you might need to reimagine a virtual machine.

To reallocate all machines in a Delivery Group

You can reallocate machines in Delivery Groups for all machine types except those created through Provisioning Services.

Desktop allocation is not available on Server OS Machines.

1. In Studio, select the Delivery Groups node.
2. Select the Delivery Group containing the machines you want to reallocate and then select Edit Delivery Group.
3. On the Machine Assignment page, add or remove the users and groups who can access any machines in the group. You can also use an import list to specify the users and groups who can access any existing or physical machines in the group. For more information, see Import and export user data.

To reallocate one machine in a Delivery Group

1. In Studio, select the Delivery Groups node.
2. Select the Delivery Group containing the machines you want to reallocate and then select View machines.
3. Select the machine you want and select Change User
4. On the Edit Assignment page, add or remove the users and groups who can access the machines.

To identify machines using tags

Tags are strings that identify machines. You can use them to refine a machine search or to limit machine access. You can add any number of tags of any length. For example, use the tags sales, marketing, or test to specify machines for different users.

You can only use tags with Desktop OS machines.

1. Select a machine. In Studio:
   - Select the Delivery Groups node, select a Desktop OS Delivery Group and then click View Machines.
   - Select Search to locate the machines.
2. Select a machine and then select Add tag.
3. Enter one or more tags, separating each tag by a semicolon (;) and then click Add tag.
4. To change or remove tags, select a machine as described in Step 1 and then select Edit tag.
5. Change the tags and then click OK.

To change the number of machines allocated to users

You can allocate more or fewer machines to the users of a Delivery Group.

1. In Studio, select the Delivery Groups node.
2. Select the Delivery Group whose users you want to provide with more or fewer machines, click Edit Delivery Group and then click End User Settings.
3. On the End User settings page, set the number of machines per user.

To update a machine's master image
1. In Studio, select the Delivery Groups node.
2. Select the Delivery Group whose machines you want to update and then click View machines.
3. Select the machine you want to update and click Update machines and then click Master image.
4. Select a snapshot from the list. To select a master image, select a snapshot and then click the plus sign to display associated master images.

To apply changes and notify machine users

1. In Studio, select the Delivery Groups node.
2. Select the Delivery Group whose machines you want to update and then click View machines.
3. Select the machine you want to update and click Update machines and then click Rollout and notification.
4. If you have made changes to the master image, select when to update the master image:
   - Immediately, restart now
   - On next restart
5. Select the restart distribution time. You can choose to restart the all machines at the same time, or time variations for rolling out the updates.
6. Select user notifications:
   - Whether to notify users about restarting the machine to which they are assigned
   - How far in advance to notify users
   - Notification message

To manage Remote PC Access machine assignments in a Delivery Group

If a machine in a Remote PC Access machine catalog is not assigned to a user, Studio temporarily assigns the machine to a Remote PC Access Delivery Group associated with the Remote PC Access machine catalog. This temporary assignment provides information to the system, so that the machine can be later assigned to an appropriate user. Along with Delivery Group assignment, Delivery Group to Machine Catalog association has a priority value. The lower the value, the higher the priority. You can only set this priority value using PowerShell Software Developer Kit (SDK) commands as described in About the XenDesktop SDK.

Priority determines to which Delivery Group a machine is assigned when:
- The unconfigured machine registers with the system
- A user needs a machine assignment

If a Remote PC Access machine catalog has multiple Delivery Group assignments, the software selects the match with the highest priority.

To remove machines from Delivery Groups

Removing a machine deletes it from a Delivery Group but does not delete the associated virtual machine from the machine catalog on which that the group is based. Therefore, the machines are available for assignment to other Delivery Groups.

You can remove machines only while they are in maintenance mode and shut down. To temporarily stop users from connecting to a machine while you are removing it, put the machine into maintenance mode before shutting it down.

Important: Machines might contain personal data. Manage this appropriately especially if the machine is allocated to another user. For example, you may need to reimagine the virtual machine.
1. In Studio, use Search to locate the machine you want to remove or select a Delivery Group and click View Machines.
2. Select the machine, and put it in maintenance mode.
3. Make sure that the machine is shut down.
4. Click Remove from Delivery Group.

To secure Delivery Groups

You can conceal all communications to and from machines in any Delivery Group using the SecureICA feature, which encrypts the ICA protocol.

When passing through public networks, Citrix recommend using additional encryption methods besides SecureICA. Citrix recommends using SSL/TLS encryption for traversing public networks. Also, SecureICA does not check data integrity.

By default, the system disables SecureICA. If you enable it, the default encryption level is 128-bit. You can configure the level using the SDK. See About the XenDesktop SDK for detailed information.

1. In Studio, select the Delivery Groups node and select the Delivery Group whose communications you want to secure.
2. Click Edit Delivery Group and then click Basic settings.
3. Select Enable Secure ICA.
Find machines, sessions, machine catalogs, applications, and Delivery Groups

Aug 11, 2014

Use the Search feature to view specific machines, sessions, machine catalogs, applications, or Delivery Groups.

Note: You cannot search within the machine catalogs or Delivery Groups tabs using the Search box. Use the Search node.

1. Select the Search node.
   - To display additional search criteria in the display, click the plus sign next to the Search drop-down fields. You can remove the search criteria by clicking the minus button.

2. Enter the name, or use the drop-down list to select another search option for the Studio item you want to find.

3. Optionally, save your search for later use by selecting Save as. The search appears in the Save Search field.

Alternatively, click the Expand Search icon (🔍) to display a drop-down list that contains search properties. Perform an advanced search by building an expression from properties in the drop-down list.

Use the following tips to enhance your search:

- To display additional characteristics to include in the display on which you can search and sort, right click any column and select Click Select columns.

- To locate a user device connected to a machine, use Endpoint and Is, and enter the device's name, or use Client (IP) and Is, and enter the device's IP address.

- To locate active sessions, use Session State, Is, and Connected.

- To list all of the machines in a Delivery Group, select the group from the Delivery Groups node and click View Machines.
Secure Delivery Groups

Mar 11, 2013
You can conceal all communications to and from machines in any Delivery Group using the SecureICA feature, which encrypts the ICA protocol.

When passing through public networks, Citrix recommend using additional encryption methods besides SecureICA. Citrix recommends using SSL/TLS encryption for traversing public networks. Also, SecureICA does not check data integrity.

By default, the system disables SecureICA. If you enable it, the default encryption level is 128-bit. You can configure the level using the SDK. See About the XenDesktop SDK for detailed information.
1. In Studio, select the Delivery Groups node and select the Delivery Group whose communications you want to secure.
2. Click Edit Delivery Group and then click Basic settings.
3. Select Enable Secure ICA.
Restrict access to machines in a Delivery Group

Mar 21, 2015

You can restrict access to a Delivery Group's machines. Any changes you make supercede previous settings, regardless of the method you use.

- Restrict access for administrators using Scopes to control administrator access to groups of objects such as machine catalogs, Delivery Groups, and Resources. You can create and assign a scope that lets administrators access all applications, and another that provides access to only certain applications.
- Restrict access for users through:
  - SmartAccess policy expressions that filter user connections made through NetScaler Gateway. Your policy administrator can perform this task in the Policy node in Studio, or through policy settings as described in Policy settings reference.
  - Exclusion filters on access policies that you set with the Software Development Kit (SDK). Access policies are applied to Delivery Groups to refine certain aspects of virtual desktop connections. For example, you can restrict machine access to a subset of the users listed on the Delivery Group's End user settings page, and you can specify the allowed user devices that can connect to machines. Access policies achieve similar results to, but are different from policies.

Using exclusion filters further refines access policies. For example, for business or security reasons you can deny access to a subset of users or devices. By default, exclusion filters are disabled and can be set using the SDK.

To restrict administrator access through scopes

1. In Studio in the Delivery Groups node, select the Delivery Group you want to restrict.
2. Click Edit Delivery Group and then click Scopes.
3. Select an existing scope.
4. Add or remove objects to include in the scope.
5. To select a object's subset, click the left-arrow to display and select sub-objects and then click OK.

To restrict user access through SmartAccess policy expressions

Use SmartAccess policy expressions through the NetScaler Gateway.

1. In Studio under Delivery Groups, select the Delivery Group you want to restrict.
2. Click Edit Delivery Group and then click Access policy.
3. On the Access Policy page, select Connections through NetScaler Gateway. Only connections through the NetScaler Gateway are allowed.
4. To choose a subset of those connections, select Connections meeting any of the following filters and:
   1. Define the NetScaler Gateway site.
   2. Add, edit, or remove the SmartAccess policy expressions that define the allowed user access scenarios for the Delivery Group. For more information about NetScaler Gateway and SmartAccess policy expressions, see Configuring SmartAccess on NetScaler Gateway.

To restrict user access through exclusion filters

You can use exclusion filters through the SDK.

In this example, there is a teaching lab on a subnet within the corporate network, and you want to prevent any access from that lab to a certain Delivery Group regardless of who is using the machines in the lab. To do so, enter the following SDK command:

```
Set-BrokerAccessPolicy -Name VPDesktops_Direct -ExcludedClientIPFilterEnabled $True

Note: You can also use the asterisk (*) as a wildcard to match all tags that start with the same policy expression. For example, if you added the tag VPDesktops_Direct to one machine and VPDesktops_Test to another, setting the tag in the Set-BrokerAccessPolicy script to VPDesktops_* applies the filter to both machines. See the About the XenDesktop SDK for more information about using the SDK.
Shut down and restart machines

Apr 17, 2013

Note: This procedure does not apply to Remote PC Access machines.

1. In Studio, use Search to locate the machine you want to shut down or restart, or select a Delivery Group and click View Machines.

2. Select the machine and perform one of the following actions. Depending on the state of the machine, some of these options are not available:
   - Force shut down — Forcibly powers off the machine and refreshes the list of machines.
   - Restart — Requests the machine's operating system to shut down and then start the machine again. If the operating system is unable to do this, the machine remains in its current state.
   - Suspend — Pauses the machine without shutting it down and refreshes the list of machines.
   - Shut down — Requests the machine's operating system to shut down.

Note: If the machine does not shut down within 10 minutes, it is powered off. If Windows attempts to install updates during shutdown, there is a risk that the machine is powered off before the updates are complete.
Remove the Shut Down command

Apr 15, 2013
Citrix recommends that you apply this Microsoft policy to all users who access Desktop OS machines.

This prevents users from selecting Shut Down within a XenDesktop session and powering off the desktop, which would require manual intervention from the system administrator.

Locate this policy at User Configuration\Administrative Templates\Start Menu and Taskbar\Remove and prevent access to the Shut Down command and set it to Enabled.
Upgrade a Delivery Group

Apr 22, 2014

Machines in a Delivery Group from an older Site might contain older Virtual Desktop Agents (VDA) versions, or run on older Windows operating systems. These include:

- VDA version: 5.0 Service Pack 1, 5.5, 5.6, 5.6 Feature Pack 1
- VDA Version 5.6 on a Microsoft Windows 7 operating system
- Machines Microsoft Windows XP or Microsoft Windows Vista operating systems

Note: Remote PC access does not support Microsoft Windows Vista.

Perform the following upgrades on machines before you upgrade the Delivery Group:

- Upgrade the operating system to Microsoft Windows 7 or Windows 8.x
- Upgrade the VDA to version 7 as described in XenDesktop 5 upgrade factors or XenDesktop 7 upgrade components.

Once you upgrade these machines' operating systems and VDA versions, upgrade their Delivery Groups to take advantage of the latest features in this release. For example, upgrading a Delivery Group turns on the ability to assign StoreFront configurations.

Note the following about machine catalog and Delivery Group versions:

- If a machine is running Windows XP or Windows Vista, or VDA version 5.6 or earlier, the machine cannot be added to a XenDesktop 7.x or XenApp Delivery Group. When creating Delivery Groups, you can select the option for Some machines are running Windows XP or Windows Vista so that these machines are in a separate Delivery Group.
- A machine with VDA version 7.x can be in any Delivery Group or machine catalog.

After upgrading the machine operating system and VDA version, and before you run Studio to upgrade the Delivery Group, you must:

- Make sure you upgrade the VDA version in the Provisioning Services console if you use Provisioning Services to create machine catalogs and Delivery Groups
- Start the machines and register them with the Delivery Controller. Otherwise, the Studio wizard cannot determine whether the machines in the Delivery Group need upgrading.

To upgrade Delivery Groups

Make sure that machines referenced by the Delivery Group have been upgraded to:

- Microsoft Windows 7 or Windows 8.x
- For machines running Windows 7, they must also run a XenDesktop 7.x or XenApp 7.5 Virtual Delivery Agent

1. In the Machines node, click Delivery Group.
2. Select the Delivery Group to upgrade. Check the Details tab, which displays operating system and VDA version information and click Upgrade Delivery Group. The wizard checks to make sure that the Delivery Group can be upgraded and indicates that the Delivery Group is suitable for upgrade.
   - If the wizard indicates that the Delivery Group is suitable for upgrade, an informational message appears. Click Upgrade and follow the prompts to complete the upgrade.
   - If one or more machines are not suitable, the wizard lists the reasons for the machine deficiencies.
     - Click View Details for detailed information. You can optionally export the information to a file by clicking Export.
Details in the Upgrade Warnings dialog box.

- Click Cancel and update the machines that need upgrades.

Citrix recommends that you resolve machine issues before upgrading the Delivery Groups to make sure that all machine function properly.

3. Configure the Delivery Group's StoreFront access.
   1. Select the Delivery Group node, select a Delivery Group, and click Edit Delivery Group.
   2. Select the StoreFront URLs to be pushed to Citrix Receiver so that Receiver can connect to a store without user intervention. You can select a StoreFront server from the display, or manually enter a StoreFront server address.

To revert a Delivery Group upgrade

You can revert machines in a Delivery Group to their prior states.

1. In the Delivery Group node, click View.
2. Select the Delivery Group to revert. Check the Details tab, which displays operating system and VDA version information and click Undo. Follow the prompts to complete the process.
Set up the restart schedule for a Server OS machine Delivery Group

May 01, 2014

Use the Reboot Schedule feature to configure the restart schedule for Server OS machines within a Delivery Group. You can:

- Specify the days and hour on which reboots occur
- Distribute the restart activity over an evenly spaced duration (this is useful for large groups)
- Create a customized message that notifies users about the restart

Note: You cannot perform an automated power-on or shutdown in Studio.

1. From the Delivery Group node, select Edit Delivery Group.
2. Click Reboot Schedule.
3. Click Yes. The Reboot Schedule options activate.
4. Enter the following information:
   - Restart machines — Select Daily, or specify a day of the week. For example, Every Saturday.
   - Restart first group at — Using 24 hour format, enter the hour and minute at which the first group restarts. For example, 20:15.
   - Restart additional groups every — Select All machines at once or the time interval at which groups restart. For example, 2 hours.
5. Select when to send a restart notification using the Send restart notification to user list. For example, 5 minutes before user is logged off. If you select this option, enter the notification message text then click OK.
Manage Server OS machine server load

Mar 28, 2015

Load Management measures the server load and determines which server to select under the current environment conditions.

When a user logs on to a Server OS machine, Load Management assigns the user to the server that is best suited to handle the request. This selection is based on:

- Server maintenance mode status
- Server load index, which is a number that represents the aggregated load, based on various measured parameters including CPU usage, memory usage, and disk usage. Load Indexes are calculated based on a formula, called the Load Evaluator.
- Concurrent logon tolerance setting, which is the allowed number of concurrent requests to log on to the server.

Maintenance mode

The server maintenance mode status and the Microsoft Windows Remote Desktop Connection (RDC) setting affect whether the Server OS machine is considered in load management.

Maintenance mode is on if any of the following occur:

- Server maintenance mode is set to On
- RDC is set to Don't allow connections to this computer
- RDC is not set to Don't allow connections to this computer, and the Remote Host Configuration User Logon Mode setting is one of the following:
  - Allow reconnections, but prevent new logons
  - Allow reconnections, but prevent new logons until the server is restarted

The Server OS machine is only considered for load balancing when maintenance mode is off.

Server load index

A server's load index determines how likely a server delivering Server OS machines is to receive connections. It is the combination of:

- The number of sessions
- Settings for performance metrics such as CPU, Disk, and Memory use

You can configure these load evaluators through the Load Management policy settings, as described in Load Management policy settings.

You can monitor this index through:

- Studio Search feature

  By default the Server Load Index column is hidden. Include this attribute in the display:

  1. Select a machine.
  2. Right-click a column heading and then choose Select Column.
  3. In the selected machine tab, scroll right to display the Machine attribute column for Server Load Index.
4. In the Machine folder, select Server Load Index.
   You can alternatively view load index information by using the PowerShell SDK Get-BrokerMachine. See for SDK information, see About the XenDesktop SDK.

   Note: A Server Load Index value of 10000 indicates that the server is at full load. If no other servers are available in the Site, users may receive a message that the desktop is currently unavailable when they launch a session.
   • Director — See the Director documentation for information about monitoring this index.

Concurrent Logon Tolerance setting

You can control the number of pending logons a server delivering Server OS machines can concurrently accept by setting the Concurrent Logon tolerance. This helps avoid server performance problems.

The Concurrent Logon Tolerance setting is equivalent to Load Throttling in XenApp.

If all servers are at or higher than the Concurrent Logons Tolerance setting, the next logon request is assigned to the server with the lowest pending logons. If more than one server meets this criteria, the server with the lowest load index is selected.
Change server load management policy settings

Feb 27, 2013
Server OS machine policies determine which settings calculate server load.

Note: This release does not support the XenApp Load Evaluator setting created for XenApp 6.5. The default Load Management settings work for most Server OS machine environments. However, if you want to change these settings, policies are available to adjust loading. See Load Management policy settings.

Servers delivering Server OS machines have default load management settings to make sure that sessions are balanced between servers. Use the Load Management policy to change the following settings.

Note: A value of -1 for any of the following settings indicates this setting is excluded from load calculation.

<table>
<thead>
<tr>
<th>Policy setting</th>
<th>Load management factor</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concurrent logons tolerance</td>
<td>The number of concurrent logons a server can accept</td>
<td>Enabled (default): Sets the maximum number of concurrent logons</td>
<td>Positive integer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Disabled: Excludes this setting from load calculation</td>
<td>(default 2)</td>
</tr>
<tr>
<td>CPU usage</td>
<td>CPU usage percentage</td>
<td>Enabled: Defines the CPU use percentage at which the server reports a full load</td>
<td>1 through 100 percent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Disabled (default): Excludes CPU usage from load calculation</td>
<td>(default 90)</td>
</tr>
<tr>
<td>CPU usage excluded process priority</td>
<td>Priorities at which the CPU usage for a process is excluded from the CPU Usage load index</td>
<td>Enabled (default): excludes processes from CPU Usage load index based on the selected value</td>
<td>Below Normal or Low — Excludes processes that have a priority or Below Normal or Low Low — Excludes processes that have a priority or Low</td>
</tr>
<tr>
<td>Disk usage</td>
<td>Disk queue length</td>
<td>Enabled: Defines the Disk queue length at which a server reports 75% full load</td>
<td>Integer value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Disabled (default): Excludes Disk usage from the load calculation</td>
<td>(default 8)</td>
</tr>
<tr>
<td>Memory</td>
<td>Memory usage</td>
<td>Enabled: Includes memory use data in load</td>
<td>1 through 100 percent</td>
</tr>
<tr>
<td>Policy setting</td>
<td>Load management factor</td>
<td>Description</td>
<td>Value</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>Memory usage base load</td>
<td>Memory usage in MBs</td>
<td>Defines the memory use in MBs up to the point at which the server reports no load due to memory usage. It is common for basic operating systems functions to consume several hundred MBs of memory.</td>
<td>1 through Installed memory in MBs (default 768)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enabled (default): Excludes an approximation of the base operating system memory usage from the server's load index</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Disabled: Does not exclude base load value from the memory usage</td>
<td></td>
</tr>
<tr>
<td>Maximum number of sessions</td>
<td>Maximum number of sessions</td>
<td>Enabled (default): Sets the maximum number of sessions on the server</td>
<td>1 through Maximum integer (default 250)</td>
</tr>
<tr>
<td></td>
<td>on a server</td>
<td>Disabled: Excludes this setting from load calculation</td>
<td></td>
</tr>
</tbody>
</table>

**To change load management policy settings**

1. In Studio, access the New Policy Wizard by selecting the Policy node.
2. Select the Categories node and then select Load Management. The load management settings described in the previous table appear.
3. Click Add to display the Edit Settings window.
4. By default, the setting is enabled. You can change the setting value.
5. To return the setting to its default, click Use default value.
Delivery Controller environment

Oct 12, 2015

In a deployment, the Delivery Controller is the server-side component that is responsible for managing user access, plus brokering and optimizing connections. Controllers also provide the Machine Creation Services that create desktop and server images.

A Site must have at least one Delivery Controller. After you install the initial Controller and create a Site, you can add additional Controllers. There are two primary benefits from having more than one Controller in a Site:

- **Redundancy** — As best practice, a production Site should always have at least two Controllers on different physical servers. If one Controller fails, the others can manage connections and administer the Site.
- **Scalability** — As Site activity grows, so does CPU utilization on the Controller and SQL Server database activity. Additional Controllers provide the ability to handle more users and more applications and desktop requests, and can improve overall responsiveness.

How Virtual Delivery Agents (VDAs) discover Controllers

Before a VDA can be used, it must register (establish communication) with a Controller on the Site. The VDA finds a Controller by checking a list of Controllers called the ListOfDDCs. The ListOfDDCs comprises one or more DNS entries or IP addresses that point the VDA to Controllers on the Site. For load balancing, the VDA automatically distributes connections across all Controllers in the list.

In addition to the ListOfDDCs, the ListOfSIDs indicates which machine Security IDs (SIDs) the VDA allows to contact it as a Controller. The ListOfSIDs can be used to decrease the load on Active Directory or to avoid possible security threats from a compromised DNS server.

It is important to ensure that the ListOfDDCs and ListOfSIDs on all VDAs contain current information as Controllers are added and removed in the Site. If the lists are not updated, a VDA might reject session launches that were brokered by an unlisted Controller. Invalid entries can delay the startup of the virtual desktop system software. To keep the lists current, you can:

- Use the auto-update feature, which automatically updates the ListOfDDCs and ListOfSIDs as Controllers are added or removed. By default, auto-update is enabled.
- Self-manage — that is, manually update policy or registry settings that identify Controllers.

Information in the ListOfDDCs and ListOfSIDs can come from several places in a deployment. The VDA checks the following locations, in order, stopping at the first place it finds the lists:

1. A persistent storage location maintained for the auto-update feature. This location contains Controller information when auto-update is enabled and after the VDA successfully registers for the first time after installation. (This storage also holds machine policy information, which ensures that policy settings are retained across restarts.)

   For its initial registration after installation, or when auto-update is disabled, the VDA checks the following locations.

2. Policy settings (Controllers, Controller SIDs).
3. The Controller information under the Virtual Desktop Agent key in the registry. The VDA installer initially populates these values, based on Controller information you specify when installing the VDA.
4. OU-based Controller discovery. This is a legacy method maintained for backward compatibility.
5. The Personality.ini file created by Machine Creation Services.

If a ListOfDDCs specifies more than one Controller, the VDA attempts to connect to them in random order. The
ListOfDDCs can also contain Controller groups, which are designated by brackets surrounding two or more Controller entries. The VDA attempts to connect to each Controller in a group before moving to other entries in the ListOfDDCs.

The auto-update feature replaces the CNAME function from earlier XenDesktop versions. You can manually re-enable the CNAME function, if desired; however, for DNS aliasing to work consistently, you cannot use both the auto-update feature and the CNAME function. See CTX137960 for information about re-enabling the CNAME functionality.

Considerations for choosing auto-update or self-manage

The policy setting that enables/disables auto-update is enabled by default.

The following types of deployments cannot use auto-update, and must self-manage.

- Deployments that use Controller groups.
- Deployments that use ListOfSIDs for security reasons. (Deployments that use ListOfSIDs to decrease the Active Directory load can use auto-update.)
- Deployments that use Provisioning Services without a write-cache disk.
- Deployments that use the Controllers or Controller SIDs policy setting.
Automatically update

Apr 22, 2013
The Enable auto update of Controllers policy setting is located in the Virtual Delivery Agent category.

- To enable auto-update, enable the Enable auto update of Controllers policy setting. This setting is enabled by default.
- To disable auto-update, disable the Enable auto update of Controllers policy setting.

When auto-update is enabled and you install a Virtual Delivery Agent (VDA), the VDA attempts to register with one of the Delivery Controller values you specified when you installed the VDA. The installer writes the Controller information you specify during VDA installation to the ListOfDDCs registry value.

After the VDA registers, the Controller with which it registered sends a list of the current Controller Fully Qualified Domain Names (FQDNs) and Security IDs (SIDs) to the VDA. The VDA writes this list to the auto-update persistent storage. Each Controller also checks the Site Configuration Database every 90 minutes for Controller information – if a Controller has been added or removed since the last check, or if a policy change has occurred, the Controller sends updated lists to its registered VDAs. The VDA will accept connections from all the Controllers in the most recent list it received.

If a VDA receives a list that does not include the Controller it is registered with (in other words, that Controller was removed from the Site), the VDA re-registers, choosing among the Controllers in the list. After a VDA registers or re-registers, it receives an updated list.

For example:

1. A deployment has three Controllers: A, B, and C. A VDA is installed and registers with Controller B (which was specified during VDA installation).
2. Two Controllers (D and E) are added to the Site. Within 90 minutes, VDAs receive updated lists and will accept connections from Controllers A, B, C, D, and E. (The load will not be spread equally to all Controllers until the VDAs are restarted.)
3. Controller B is removed from the Site. Within 90 minutes, VDAs receive updated lists because there has been a Controller change since the last check. The VDA installed in step 1 is registered with Controller B, which is no longer on the list, so that VDA re-registers, choosing among the Controllers in the current list (A, C, D, and E).
Self-manage

Feb 05, 2013

If you do not use auto-update, you must update the Citrix policy setting or registry values for each Virtual Delivery Agent (VDA) in the site (or the VDA image) after you add, move, or remove Delivery Controllers in the Site. Registry changes can also be updated using Group Policy Object.

To self-manage using Citrix policy settings

Update the FQDN values specified in the Controllers policy setting. This policy setting is located in the Virtual Delivery Agent category.

If you also use ListOfSIDs in your deployment, update the SID values specified in the Controller SIDs policy setting.

To self-manage using the registry

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

Update the ListOfDDCs registry key, which lists the FQDNs of all the Controllers in the Site. (This key is the equivalent of the Active Directory Site OU.) Separate multiple values with spaces. Surround Controller groups with brackets.

HKEY_LOCAL_MACHINE\Software\Citrix\VirtualDesktopAgent\ListOfDDCs (REG_SZ)

If the HKEY_LOCAL_MACHINE\Software\Citrix\VirtualDesktopAgent registry location contains both the ListOfDDCs and FarmGUID keys, ListOfDDCs is used for Controller discovery; FarmGUID is present if a site OU was specified during VDA installation.

Optionally, update the ListOfSIDs registry key:

HKEY_LOCAL_MACHINE\Software\Citrix\VirtualDesktopAgent\ListOfSIDs (REG_SZ)
Add, remove, or move Controllers

Feb 07, 2014

To add, remove, or move a Delivery Controller, you need the following roles or permissions:

<table>
<thead>
<tr>
<th>Operation</th>
<th>Purpose</th>
<th>Server role</th>
<th>Database role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database creation</td>
<td>Create suitable empty database</td>
<td>dbcreator</td>
<td></td>
</tr>
<tr>
<td>Schema creation</td>
<td>Create all service-specific schemas and add first Controller to Site</td>
<td>securityadmin</td>
<td>db_owner</td>
</tr>
<tr>
<td>Add Controller</td>
<td>Add Controller (other than the first) to the Site</td>
<td>securityadmin</td>
<td>db_owner</td>
</tr>
<tr>
<td>Add Controller (mirror server)</td>
<td>Add Controller login to the database server currently in the mirror role of a mirrored database</td>
<td>securityadmin</td>
<td></td>
</tr>
<tr>
<td>Remove Controller</td>
<td>Remove Controller from the Site</td>
<td>db_owner</td>
<td></td>
</tr>
<tr>
<td>Schema update</td>
<td>Apply schema updates or hotfixes</td>
<td>db_owner</td>
<td></td>
</tr>
</tbody>
</table>

* While technically more restrictive, in practice, the securityadmin server role should be treated as equivalent to the sysadmin server role.

When using Desktop Studio to perform these operations, the user account must explicitly be a member of the sysadmin server role.

Before adding, removing, or moving a Controller, if your deployment uses database mirroring, ensure that the principal and mirrored databases are both running. In addition, if you are using scripts with SQL Server Management Studio, enable SQLCMD mode before executing the scripts. To verify mirroring after adding, removing, or moving a Controller, run the get-configdbconnection PowerShell cmdlet to ensure that the Failover Partner has been set in the connection string to the mirror.

After you add, remove, or move a Controller:

- If auto-update is enabled, the Virtual Delivery Agents (VDAs) will receive an updated list of Controllers within 90 minutes.
- If auto-update is not enabled, ensure that the Controller policy setting or ListOfDDCs registry key are updated for all VDAs. After moving a Controller to another Site, update the policy setting or registry key on both Sites.

To add a Controller

You cannot add servers installed with an earlier version of this software to a Site that was created with this version.

1. On the server you want to add, run the installer and select the Delivery Controller and any other core components you want to install.

2. In Studio, click Join existing deployment and enter the Site address.

To remove a Controller
Removing a Controller does not uninstall the Citrix software or any other component; it removes the Controller from the Database so that it can no longer be used to broker connections and perform other tasks. If you remove a Controller, you can later add it back to the same Site or to another Site. A Site requires at least one Controller, so you cannot remove the last one listed in Studio.

Note: Make sure that the Controller is powered on so that Studio loads in less than one hour. Once Studio loads the Controller you want to remove, power off the Controller when prompted to do so.

When you remove a Controller from a Site, the Controller logon to the database server is not removed. This avoids potentially removing a logon that is used by other products' services on the same machine. The logon must be removed manually if it is no longer required; securityadmin server role permission is needed to remove the logon.

Do not remove the Controller from Active Directory until after you remove it from the Site.

1. In Studio, select Configuration > Controllers in the left pane, then select the Controller you want to remove.
2. Click Remove Controller in the Actions pane. If you do not have the correct database roles and permissions, you are offered the option of generating a script that allows your database administrator to remove the Controller for you.
3. You might need to remove the Controller's machine account from the database server. Before doing this, check that another service is not using the account.

After using Studio to remove a Controller, traffic to that Controller might linger for a short amount of time to ensure proper completion of current tasks. If you want to force the removal of a Controller in a very short time, Citrix recommends you shut down the server where it was installed, or remove that server from Active Directory. Then, restart the other Controllers on the Site to ensure no further communication with the removed Controller.

To move a Controller to another Site

You cannot move a Controller to a Site that was created with an earlier version of this software.

1. On the Site where the Controller is currently located (the old Site), in Studio, select Configuration > Controllers in the left pane, then select the Controller you want to move.
2. Click Remove Controller in the Actions pane. If you do not have the correct database roles and permissions, you are offered the option of generating a script that allows your database administrator to remove the Controller for you. A Site requires at least one Controller, so you cannot remove the last one listed in Studio.
3. On the Controller you are moving, open Studio, reset the services when prompted, click Join existing site, and enter the address of the new site.
Move a Virtual Delivery Agent (VDA) to another Site

Feb 24, 2014

You can move a VDA to another Site (from Site 1 to Site 2) when upgrading, or when moving a VDA image that was created in a test Site to a production Site. There are two ways to do this: using the installer or Citrix policies.

Move a VDA using the installer

Important: Specify Delivery Controllers in the installer only when the Controllers policy setting is not used.
Run the installer and add a Delivery Controller, specifying the FQDN (DNS entry) of a Controller in Site 2.

Move a VDA using Citrix policies

Use the Group Policy Editor to move multiple VDAs between Sites.

For example:

1. Create a policy in Site 1 that specifies the following settings, then filter the policy to the Delivery Group level to initiate a staged VDA migration between the Sites.
   - Controllers - containing FQDNs (DNS entries) of one or more Controllers in Site 2.
   - Enable auto update of Controllers - set to disabled.
2. Each VDA in the Delivery Group is alerted within 90 minutes of the new policy. The VDA ignores the list of Controllers it receives (because auto-update is disabled); it selects one of the Controllers specified in the policy, which lists the Controllers in Site 2.
3. When the VDA successfully registers with a Controller in Site 2, it receives the Site 2 ListOfDDCs and policy information, which has auto-update enabled by default. Since the Controller with which the VDA was registered in Site 1 is not on the list sent by the Controller in Site 2, the VDA re-registers, choosing among the Controllers in the Site 2 list. From then on, the VDA is automatically updated with information from Site 2.
Active Directory OU-based Controller discovery

Feb 05, 2013

This Delivery Controller discovery method is supported primarily for backward compatibility, and is valid only for Virtual Delivery Agents (VDAs) for Windows Desktops, not VDAs for Windows Servers. Active Directory-based discovery requires that all computers in a Site are members of a domain, with mutual trusting relationships between the domain used by the Controller and the domain(s) used by desktops. If you use this method, you must configure the GUID of the OU in each desktop registry.

To perform an OU-based Controller discovery, run the Set-ADControllerDiscovery.ps1 PowerShell script on the Controller (each Controller contains this script in the folder $Env:ProgramFiles\Citrix\Broken\Service\Setup Scripts). To run the script, you must have CreateChild permissions on a parent OU, plus full administration rights.

When you create a Site, a corresponding Organizational Unit (OU) must be created in Active Directory if you want desktops to discover the Controllers in the Site through Active Directory. The OU can be created in any domain in the forest that contains your computers. As best practice, the OU should also contain the Controllers in the Site, but this is not enforced or required. A domain administrator with appropriate privileges can create the OU as an empty container, then delegate administrative authority over the OU to a Citrix administrator.

The script creates several essential objects. Only standard Active Directory objects are created and used. It is not necessary to extend the schema.

- A Controllers security group. The computer account of all Controllers in the Site must be a member of this security group. Desktops in a Site accept data from Controllers only if they are members of this security group. Ensure that all Controllers have the ‘Access this computer from the network’ privilege on all virtual desktops running the VDA. You can do this by giving the Controllers security group this privilege. If Controllers do not have this privilege, VDAs will not register.

- A Service Connection Point (SCP) object that contains information about the Site, such as the Site name. If you use the Active Directory Users and Computers administrative tool to inspect a Site OU, you might need to enable Advanced Features in the View menu to see SCP objects.

- A container called RegistrationServices, which is created in the Site OU. This contains one SCP object for each Controller in the Site. Each time the Controller starts, it validates the contents of its SCP and updates it, if necessary.

If multiple administrators are likely to add and remove Controllers after the initial installation, they need permissions to create and delete children on the RegistrationServices container, and Write properties on the Controllers security group; these permissions are granted automatically to the administrator who runs the Set-ADControllerDiscovery.ps1 script. The domain administrator or the original installing administrator can grant these permissions, and Citrix recommends setting up a security group to do this.

When you are using a Site OU:

- Information is written to Active Directory only when installing or uninstalling this software, or when a Controller starts and needs to update the information in its SCP (for example, because the Controller was renamed or because the communication port was changed). By default, the Set-ADControllerDiscovery.ps1 script sets up permissions on the objects in the Site OU appropriately, giving each Controller Write access to its SCP. The contents of the objects in the Site OU are used to establish trust between desktops and Controllers. Ensure that:

  - Only authorized administrators can add or remove computers from the Controllers security group, using the security group's access control list (ACL).
Only authorized administrators and the respective Controller can change the information in the controller's SCP.

If your deployment uses replication, be aware of potential delays; see the Microsoft documentation for details. This is particularly important if you create the Site OU in a domain that has domain controllers in multiple Active Directory sites. Depending on the location of desktops, Controllers, and domain controllers, changes that are made to Active Directory when you are initially creating the Site OU, installing or uninstalling Controllers, or changing Controller names or communication ports might not be visible to desktops until that information is replicated to the appropriate domain controller. The symptoms of such replication delay include desktops that cannot establish contact with Controllers and are therefore not available for user connections.

This software uses several standard computer object attributes in Active Directory to manage desktops. Depending on your deployment, the machine object’s fully qualified domain name, as stored in the desktop’s Active Directory record, can be included as part of the connection settings that are returned to the user to make a connection. Ensure that this information is consistent with information in your DNS environment.

Follow the directions in To move a controller. After you remove the Controller from the old Site (step 2), run the PowerShell script: Set-ADControllerDiscover –sync.

This script synchronizes the OU with the current set of Controllers. After joining the existing Site (step 4), run the same script on any Controller in the new Site.
Use SSL on Controllers and change HTTP/HTTPS ports

Nov 05, 2013
The XML Service runs on the Delivery Controller and supports the HTTP and HTTPS protocols.

For HTTPS, the XML Service supports Secure Sockets Layer (SSL) features through the use of server certificates but not client certificates. To use HTTPS, you must obtain, install, and register a server certificate on all Controllers, and configure a port with the SSL certificate.

- If the Controller has IIS installed, use the steps described in http://support.microsoft.com/kb/299875.
- If the Controller does not have IIS installed, one method of configuring the certificate is:
  2. Configure a port with the certificate; see http://msdn.microsoft.com/en-us/library/ms733791(v=vs.110).aspx.

By default, the XML Service listens on port 80 for HTTP traffic and port 443 for HTTPS traffic. Although you can use non-default ports on a Controller for HTTP or HTTPS traffic, be aware of the security risks of exposing a Controller to untrusted networks. Instead of changing the defaults, it is preferable to deploy a standalone StoreFront server.

1. Run the following command on the Controller: BrokerService.exe -WIPORT http port -WISSLPORT https port
   http port is the port number for HTTP traffic and https port is the port number for HTTPS traffic.

2. If you want the XML Service to ignore HTTP or HTTPS traffic on the default ports, set the following registry values on the Controller and restart the Broker Service. Both values are located in HKLM\Software\Citrix\DesktopServer. Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.
   - To ignore HTTP traffic, set XmlServicesEnableNonSsl to 0.
   - To ignore HTTPS traffic, set XmlServicesEnableSsl to 0.
Change secondary database locations

By default, the Configuration Logging and Monitoring databases (the secondary databases) are located on the same server as the Site Configuration database. Initially, all three databases have the same name. Citrix recommends that you change the location of the secondary databases after you create a Site. You can host the Configuration Logging and Monitoring databases on the same server or on different servers. The backup strategy for each database may differ.

When you change the location of the Configuration Logging or Monitoring database:
- The data in the previous database is not imported to the new database.
- Logs cannot be aggregated from both databases when retrieving logs.
- The first log entry in the new database indicates that a database change occurred, but it does not identify the previous database.

Before you change the location of the Configuration Logging or Monitoring database, install a supported version of Microsoft SQL Server on the server where the database will reside. Set up mirror, cluster, or other supported redundancy infrastructures, as needed.

You cannot change the location of the Configuration Logging database when mandatory logging is enabled.

Note: You cannot use this method to change the location of the Site Configuration database.

1. From Studio, select Configuration in the left pane. The names and addresses of the three databases are listed, plus mirror server addresses, if configured.
2. Select the database for which you want to specify a new location, then click Change Database in the Actions pane.
3. In the Change Database dialog box, specify the location of the server containing the new SQL Server installation (using one of the forms in the following table) and the database name.

<table>
<thead>
<tr>
<th>Database type</th>
<th>What to enter</th>
<th>With this database configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standalone or mirror</td>
<td>servername</td>
<td>The default instance is used and SQL Server uses the default port.</td>
</tr>
<tr>
<td></td>
<td>Servername\INSTANCENAME</td>
<td>A named instance is used and SQL Server uses the default port.</td>
</tr>
<tr>
<td></td>
<td>servername,port-number</td>
<td>The default instance is used and SQL Server uses a custom port. (The comma is required.)</td>
</tr>
<tr>
<td>Other</td>
<td>cluster-name</td>
<td>A clustered database.</td>
</tr>
<tr>
<td></td>
<td>availability-group-listener</td>
<td>An Always-On database.</td>
</tr>
</tbody>
</table>

4. If you want Studio to create the database, click OK. When prompted, click OK, and Studio will create the database automatically. Studio attempts to access the database using the current Studio user's credentials; if that fails, you are prompted for the database user's credentials. Studio then uploads the database schema to the database. (The credentials are retained only for the database creation time frame.)
5. If you want to create the database manually, click Generate script. The generated scripts includes instructions for
manually creating the database and a mirror database, if needed. Ensure that the database is empty and that at least one user has permission to access and change the database before uploading the schema.
Connections and resources

Dec 30, 2013
You must be a Full Administrator to complete these tasks. Read Only Administrators can view Connection, resource, and machine details.

For information about deploying this product with supported host systems, see the — Integrate documentation.

Hosts must be present when creating a Connection.

1. In Studio, select Configuration > Hosting.
2. Click Add Connections and Resources.
3. Select Create a new Connection.
4. Enter a Connection name.
5. Specify the type of host, its address, and the credentials to use when connecting. Make sure that the credentials let you perform all the necessary tasks. If you use XenServer:
   • Citrix recommends using HTTPS to secure communication with XenServer. To use HTTPS you must replace the default SSL certificate installed with XenServer with one from a trusted certificate authority; see CTX128656.
   • You can later select the hypervisors to be used for high availability if it is enabled on XenServer. To select hypervisors, finish creating the Connection, select it, and click Change details. Citrix recommends that you select all servers in the pool to allow communication with XenServer if the pool master fails.
   If you manage user desktops hosted on dedicated blade PCs in the data center, set Host type to None.
6. Select Studio Tools (Machine Creation Services) or Other tools to create VMs. To select hypervisors that provide GPU resources, you must choose Machine Creation Services (MCS).
7. When using MCS, select the network and storage resources for the new virtual machines.
   • If you use shared storage, you can enable IntelliCache to reduce load on the storage device. Local storage is on the Host. For more information about using IntelliCache, see Use IntelliCache with XenDesktop
8. If your Connection has GPU capabilities:
   1. Select Yes for Do you want to use graphics virtualization?
   2. Select a GPU type and group from the list.
9. Enter a name for the resources and then click Finish.

1. In Studio, select Configuration > Hosting.
2. Click Add Connections and Resources.
3. Select Use an existing Connection, select the relevant Connection from the list.
4. Select Machine Creation Services (MCS) or Other tools and then click Next. To select hypervisors that provide GPU resources, you must choose Machine Creation Services (MCS). If your connection has GPU capabilities:
   1. Select Yes for Do you want to use graphics virtualization?
   2. Select a GPU type and group from the list.
5. Enter a name for the resources and then click Finish.
1. In Studio, select Configuration > Hosting.
2. Select the resource to which you want to add storage, then click Add storage.
3. Select the storage to add and then click OK.

1. In Studio, select Configuration > Hosting.
2. Select the Connection you want to edit and then click Change details.
3. Edit the Connection:
   - To update the Connection address and credentials, click Edit settings, enter the new details, and then click OK. Follow the steps in
     — To create a Connection and resources
   
   Note: To rename the Connection, select it and then click Rename.
   - To specify the high availability servers (enabled on XenServer), click Edit HA servers. Citrix recommends that you select all servers in the pool to allow communication with XenServer if the pool master fails.
   - To configure hypervisor throttling, click Advanced. If your power management settings allow too many or too few machines to start at the same time, you can adjust the throttling limit as follows:
     - To prevent more than a certain number of operations or actions running at any one time, select a number for Max active actions.
     - To limit the number of new actions that can be started per minute, select a number for Max new actions per minute.
     - To limit the number of concurrent actions to a percentage of the total number of VMs configured for this Connection, select a number for Max power actions as a percentage of desktops.
     - To limit the number of Personal vDisk power actions as a percentage, select a number for Max Personal vDisk power actions as a percentage.
     The actual limit applied is the lower number of the configured settings. For example, if the maximum active number of actions is 10, the maximum number of actions as a percentage of desktops is 10, and the number of machines is 34, the limit is 3 (that is, 10% of 34 rounded to the nearest whole number).
     
     Note: Use Connection options only under the guidance of a Citrix Support representative.
   - To change the scopes for the Connection, click Scopes.
   - You cannot change the GPU settings for an existing connection, because machine catalogs accessing this resource must use an appropriate GPU-specific master image. You must create a new Connection as described in
     — To create a Connection and resources
     
Placing a Connection into maintenance mode prevents any new power action from affecting any machine stored on the Connection. No user can connect to a machine in this state. If a user is already connected, maintenance mode takes effect as soon as they log off. You can then perform administrative tasks on the associated image, such as applying patches and upgrades using your image management tools.

1. In Studio, select Configuration > Hosting.
2. Select the Connection to put into maintenance mode and then click Turn on maintenance mode. To take a Connection out of maintenance mode, click Turn off Maintenance Mode.
Before you delete a Connection, make sure that:

- All users have logged off from the machines stored on the Connection
- No disconnected user sessions are still running
- For pooled and dedicated machines, all machines are in maintenance mode
- For existing Machine Catalogs, all machines are powered off

Caution: Deleting a Connection can result in the deletion of large numbers of machines and in loss of data. Make sure you read this topic carefully and that any user data on affected machines is backed up or no longer required.

1. In Studio, select Configuration > Hosting.
2. Select the Connection you want to delete and then click Delete Connection.
3. If this Connection still has machines stored on it, you are prompted to specify whether or not the machines should be deleted and, if they are to be deleted, what should be done with the Active Directory computer accounts associated with them. A catalog becomes unusable when you delete a Connection that is referenced by that catalog. If this Connection is referenced by a catalog you are given the opportunity to delete the catalog at this point. Before you delete a catalog, make sure that it is not supported by other Connections.

1. In Studio, select Configuration > Hosting.
2. Select the Connection you want to rename and then click Rename Connection.
3. Enter the new name and then click OK.

1. In Studio, select Configuration > Hosting.
2. Select the relevant Connection.
3. Select View Machines. The machines accessed through this Connection are listed in the upper panel of the window. To display the details of a machine, select it, and the details appear in the lower panel. Session details are also provided if there is a session open.

To find machines quickly, use the search features. Either select a saved search from the list at the top of the window or create a new search. You can either search by typing all or part of the machine name or you can build an expression to use for an advanced search. To build an expression, click the unfold button and then select from the lists of properties and operators.

1. Display the machines by selecting Configuration > Hosting.
2. Select the relevant machines.
3. Select one of the following actions:
   - Start. Starts the machine if it is powered-off or suspended. If the Host type does not support the power-on function, the Start action is not available.
   - Suspend. Pauses the desktop without shutting it down and refreshes the list of desktops.
   - Shut down. Requests the desktop's operating system to shut down.
     Note: If the desktop does not shut down within 10 minutes, it is powered off. If Windows attempts to install updates during shutdown, there is a risk that the desktop will be powered off before the updates are complete.
   - Force shut down. Forcibly powers off the desktop and refreshes the list of desktops.
   - Restart. Requests the desktop operating system to shut down and then start the desktop again. If the operating system is unable to do this, the desktop remains in its current state.
   - Enable maintenance mode. To temporarily stop connections to a machine so that maintenance tasks can be carried out.
out, put it into maintenance mode. Users cannot connect to a machine in this state. If a user is already connected, maintenance mode takes effect as soon as they log off.

Note: To put all the machines accessed through a Connection into maintenance mode, select the Connection and click Turn on Maintenance Mode, as described in "To place a Connection into maintenance mode" above.

- Remove from Delivery Group. Removing a machine from a Delivery Group does not delete it from the catalog on which the group is based. You can remove a machine only while no user is connected to it. To temporarily stop users from connecting to a machine while you are removing it, put the machine into maintenance mode first. Users cannot connect to a machine that is removed from a Delivery Group.

- Delete. When you delete a machine, users no longer have access to it and the machine is deleted from the catalog. Before deleting a machine, make sure all user data is backed up or no longer required. You can delete a machine only while no user is connected to it. To temporarily stop users from connecting to a machine while you are deleting it, put the machine into maintenance mode first.

1. In Studio, select Configuration > Hosting.
2. Select the resource you want to delete and then click Delete Resources.

1. In Studio, select Configuration > Hosting.
2. Select the resource you want to rename and then click Rename Resources.
IntelliCache with XenDesktop 7

Jul 17, 2013

Using IntelliCache, hosted VDI deployments are more cost-effective as this enables you to use a combination of shared storage and local storage. Performance is enhanced, and network traffic is reduced. The local storage caches the master image from the shared storage, which reduces the amount of reads on the shared storage. For shared desktops, writes to the differencing disks are written to local storage on the host and not to shared storage.

Your shared storage must be NFS if you are using IntelliCache.

Citrix recommends that you use a high performance local storage device to ensure the fastest possible data transfer.

To use IntelliCache you must enable it in both XenServer and XenDesktop.

To enable IntelliCache in XenServer

When installing XenServer, select Enable thin provisioning (Optimized storage for XenDesktop). Citrix does not support mixed pools of servers that have IntelliCache enabled and servers that do not.

For more information on using IntelliCache, see the XenServer and IntelliCache chapter in the XenServer Installation Guide available from the XenServer node.

To enable IntelliCache in XenDesktop

IntelliCache is disabled by default in XenDesktop. You can enable it when you are adding a host, provided the XenServer pool is XenServer 5.6 Service Pack 2 or later. You can update the setting only when the host is created; you cannot disable IntelliCache at a later date. The option to enable IntelliCache is not available if you are configuring a site using Quick Deploy.

1. When you are adding a XenServer host and you are prompted for the type of storage to use, select Shared.
2. Select Use IntelliCache to reduce load on the shared storage.
Manage Citrix policies

Feb 27, 2014
To control user access or session environments, configure a Citrix policy. Citrix policies are the most efficient method of controlling connection, security, and bandwidth settings.

You can create policies for specific groups of users, devices, or connection types. Each policy can contain multiple settings.

You work with policies through Studio or the Group Policy Management Console in Windows. The console or tool you use to do this depends on whether or not your network environment includes Microsoft Active Directory and whether or not you have the appropriate permissions to manage Group Policy Objects (GPOs).

If you are a Citrix administrator without permission to manage Group Policy, use Studio to create policies for your site. Policies created using Studio are stored in the site database and updates are pushed to the virtual desktop either when that virtual desktop registers with the broker or when a user connects to that virtual desktop.

If your network environment includes Active Directory and you have the appropriate permissions to manage Group Policy, you may want to use the Group Policy Editor to create policies for your site. The settings you configure affect the GPOs you specify through the Group Policy Management Console.

Important: You must use Group Policy Editor to configure some policy settings, including:

- Policy settings relating to registering virtual desktops with a controller.
- Policy settings relating to Microsoft Application Virtualization (App-V) servers.

Group Policy settings are processed in the following order:
1. Local GPO
2. XenDesktop site GPO (stored in the site database)
3. Site-level GPOs
4. Domain-level GPOs
5. Organizational Units

However, in the event of a conflict, policy settings that are processed last can overwrite those that are processed earlier. This means that policy settings take precedence in the following order:
1. Organizational Units
2. Domain-level GPOs
3. Site-level GPOs
4. XenDesktop site GPO (stored in the site database)
5. Local GPO

For example, a Citrix administrator creates a policy (Policy A) through Studio that enables client file redirection for the company's sales employees. Meanwhile, another administrator creates a policy (Policy B) through Group Policy Editor that disables client file redirection for the sales employees. When the sales employees log on to their virtual desktops, Policy B is applied and Policy A is ignored. This happens because Policy B was processed at the domain level and Policy A was processed at the XenDesktop site GPO level.
Note, however, that when a user launches an ICA or Remote Desktop Protocol (RDP) session, Citrix session settings override the same settings configured in an Active Directory policy or using Remote Desktop Session Host Configuration. This includes settings that are related to typical RDP client connection settings such as Desktop wallpaper, Menu animation, and View window contents while dragging.

The process for configuring policies is as follows:

1. Create the policy.
2. Configure policy settings.
3. Assign the policy to machine and user objects.
4. Prioritize the policy.
5. Verify the effective policy by running the Citrix Group Policy Modeling wizard.
Navigate Citrix policies and settings

Apr 03, 2013
In Studio, policies and templates are located under the Policy node.

Within policies and templates, policy settings are sorted into categories, based on the functionality or feature they affect. For example, policy settings relating to Profile management are all located in the Profile management section.

Policy settings can apply to either machines or users. In Studio, Computer settings (policy settings applying to machines) define the behavior of virtual desktops and are applied when a virtual desktop starts. Note that these settings apply even when there are no active user sessions on the virtual desktop. User settings define the user experience when connecting to virtual desktops using ICA. User policies are applied whenever a user connects or reconnects to a virtual desktop using ICA. If a user connects to a virtual desktop using RDP or logs on directly at the console, user policies are not applied. When policies are applied within your environment, the system determines which settings relate to machines and which relate to users and applies those settings accordingly.

Active Directory policies and settings are collected into two categories: Computer Configuration and User Configuration.

In Studio you access policies and settings by selecting the Policy node in the left pane of the console and selecting either the Policies or Templates tab.

In the Group Policy Editor, you access policies and settings by selecting the Citrix Policies node under Computer Configuration or User Configuration in the tree pane.

In Studio, the Policies tab displays a list of all policies that have been created. To the right of this list, the following tabs are displayed:
- Overview displays the high-level details for the selected policy, including policy name, priority, and whether or not the policy is currently enabled or disabled.
- Settings displays a list of all configured settings for the selected policy.
- Assigned to displays a list of all user and machine objects to which the selected policy is currently assigned.

The Templates tab displays lists of both Citrix-provided templates and custom templates defined by you. To the right of this list, the following tabs are displayed:
- Description displays a description for the selected template and provides information about why you may want to use the template to create a policy in your environment.
- Settings displays a list of all configured settings for the selected template.

You can search for settings in the templates and policies you create. All searches find items by name as you type. To search for a setting within a policy or template:
1. Select the policy or template you want to search.
2. Select Edit policy ... or Edit Template...
3. On the Settings page, begin to type the name of the setting you are searching for.

You can further refine your search by:
- Selecting a specific product version from the drop-down list, to search only the settings relating to that product version.
- Selecting the View selected only check box or selecting to search only the settings that have been added to the
selected policy.

- Selecting a category such as Auto Client Reconnect or Bandwidth to search only the settings in that category.

To search the entire catalog of settings, select All Settings.
Manage Citrix policy templates

Apr 10, 2013

Policy templates allow you to configure Citrix policies to effectively manage the end user experience within your environment. Templates consist of pre-configured settings that optimize performance for specific environments or network conditions, for example when there is a low bandwidth or users require a high quality user experience. Policies can be applied to server and desktop operating systems or to individual user sessions. You can use templates in the following ways:

- As a source for creating other policies
- As a tool with which to compare existing policies
- As a method for delivering or receiving policy configurations from Citrix Support or trusted third parties

You can perform the following tasks with policy templates:

- Create new templates using existing templates or policies
- Create new policies using existing templates
- Import and export templates

Policy templates are displayed on the Templates tab in Studio and Group Policy Editor. In Studio, templates are displayed in a single list. In Group Policy Editor, Computer templates are displayed when you are working with Computer policies. Likewise, User templates are displayed when you are working with User policies.

When selected, each template displays the following information tabs beneath the templates list:

- Description displays a concise overview of the purpose of the selected template.
- Settings displays a list of all the configured settings, their current values, and their default values in the selected template.

Policy templates are local files that are stored on the machine running Studio and not in the site database. Local files are controlled by Windows administrative permissions rather than Delegated Administration roles and scopes. This means that an administrator granted read-only permissions under the Delegated Administration model can, for example, create new templates. Because templates are local files, however, no changes are actually made to your environment.

Citrix provides the following built-in templates:

- High Definition User Experience templates include settings for providing high quality audio, graphics, and video to users.
- High Server Scalability templates include settings for providing an optimized user experience while hosting more users on a single server.
- Optimized Bandwidth for WAN templates include settings for providing an optimized experience to users with low bandwidth or high latency connections; for example, users working from branch offices over a shared WAN connection.
- Security and Control templates include settings for disabling access to peripheral devices, drive mapping, port redirection, and Flash acceleration on user devices.

You can use these templates as a model for creating new policies or templates. Built-in templates are created and updated by Citrix. You cannot modify or delete these templates. You can, however, modify or delete templates that you create or import through Studio or Group Policy Editor.
Create policy templates

Jun 17, 2013

You create templates from either an existing template or an existing policy. The new template is then populated with the same settings as the original template or policy. Any assignments specified in the original policy are not included in the template.

1. From Studio, select the Policy node in the left pane.
2. Click the Templates tab and then select the template from which you want to create the new template.
4. Select and configure the policy settings you want to include in the template. Remove any existing settings that should not be included.
5. Enter a name for the template.
6. Click Finish. The new template appears on the Templates tab.

1. From Studio, select the Policy node in the left pane.
2. On the Policies tab, select the policy from which you want to create the template.
3. Click Save as Template. The Save as Template dialog box appears.
4. Select and configure any new policy settings you want to include in the template. Remove any existing settings that should not be included.
5. Enter a name and a description for the new template.
6. Click Finish. The new template appears on the Templates tab.
Import and export policy templates

Jun 17, 2013

Policy templates are local to the computer on which you are running the console to manage your site. You can transfer policy configurations between environments, including other sites that you manage on the computer running the console.

You transfer templates by importing or exporting them. This allows you to:
- Implement policy configurations from servers in other sites
- Create backups of your template files to aid recovery of policy configurations
- Supply policy configurations from your site to aid Citrix Support in troubleshooting issues
- Implement policy configurations created by Citrix Support to resolve issues in your site

1. From Studio, select the Policy node in the left pane.
2. Click the Templates tab and then click Import Template. The Import Template dialog box appears.
3. Select the template file you want to import and click Open. The imported template appears in the templates list.
   Note: If you import a template with the same name as an existing template, you can choose to overwrite the existing template or save the template with a different name that is generated automatically.

1. From Studio, select the Policy node in the left pane.
2. Click the Templates tab and then select the template you want to export.
3. Click Export Template. The Export Template dialog box appears.
4. Select the location where you want to save the template and click Save.

A .gpt file is created in the location you specified.
Create policies

Jun 17, 2013

Before you create a policy, decide which group of users or devices you want it to affect. You may want to create a policy based on user job function, connection type, user device, or geographic location. Alternatively, you can use the same criteria that you use for Windows Active Directory group policies.

If you already created a policy that applies to a group, consider editing the policy and configuring the appropriate settings instead of creating another policy. Avoid creating a new policy solely to enable a specific setting or to exclude the policy from applying to certain users.

You can create policies using the following methods:

- Create a new policy based on the settings included in a policy template
- Create a new policy using the New Policy wizard

By default, the new policy includes all the same settings as the original template. You can choose to accept these default settings or to customize the policy according to your needs.

1. From Studio, select the Policy node in the left pane.
2. Click the Templates tab and select the template from which you want to create the new policy.
3. Click Create Policy... The Create Policy wizard appears.
4. Choose whether or not to customize the policy. If you choose not to customize the policy, proceed to Step 6. If you choose to customize the policy, click Modify defaults and add more settings, and then add or remove settings, as required.
5. Choose how to apply the policy. Do one of the following:
   - Click Assign to selected user and machine objects and then choose the user and machines objects to which you want to apply the policy. This option applies the policy only to the user and machine objects you selected.
   - Click Assign to all objects in a site. This option applies the policy to all user and machine objects in your site.
6. Enter a unique name for the new policy or accept the default name that is generated automatically. Optionally, provide a description.
   - Note: Consider naming the policy according to who or what it affects; for example, Accounting Department or Remote Users.
7. Leave the policy enabled or clear the Enable policy checkbox to disable the policy. Enabling the policy allows it to be applied immediately to users logging on to the site. Disabling the policy prevents it from being applied. If you need to prioritize the policy or add settings at a later time, consider disabling the policy until you are ready to apply it to users.

The New Policy wizard enables you to create a new policy to which you can add the settings you require.

1. From Studio, select the Policy node in the left pane.
2. Click Create Policy Manually. The Create Policy wizard appears.
3. Add and configure individual policy settings, as required.
4. Choose how to apply the policy. Do one of the following:
   - Click Assign to selected user and machine objects and then choose the user and machines objects to which you want
to apply the policy. This option applies the policy only to the user and machine objects you selected.

- Click Assign to all objects in a site. This option applies the policy to all user and machine objects in your site.

5. Enter a unique name for the new policy or accept the default name that is generated automatically. Optionally, provide a description.
   
   Note: Consider naming the policy according to who or what it affects; for example, Accounting Department or Remote Users.

6. Leave the policy enabled or clear the Enable policy checkbox to disable the policy. Enabling the policy allows it to be applied immediately to users logging on to the site. Disabling the policy prevents it from being applied. If you need to prioritize the policy or add settings at a later time, consider disabling the policy until you are ready to apply it to users.
Configure policy settings

Jan 28, 2013

Policies contain settings that are applied to connections when the policy is enforced. Policy settings can be enabled, disabled, or not configured. By default, policy settings are not configured, meaning they are not added to a policy. Settings are applied only when they are added to a policy.

Some policy settings can be in one of the following states:
- Allowed or Prohibited allows or prevents the action controlled by the setting.
- Enabled or Disabled turns the setting on or off. If you disable a setting, it is not enabled in lower-ranked policies.

For settings that are Allowed or Prohibited, the action controlled by the setting is either allowed or prevented. In some cases, users are allowed or prevented from managing the setting's action in a session. For example, if the Menu animation setting is set to Allowed, users can control menu animations in their client environment.

In addition, some settings control the effectiveness of dependent settings. For example, the Client drive redirection setting controls whether or not users are allowed to access the drives on their devices. To allow users to access their network drives, both this setting and the Client network drives setting must be added to the policy. If the Client drive redirection setting is disabled, users cannot access their network drives even if the Client network drives setting is enabled.

In general, policy setting changes that impact machines go into effect either when the virtual desktop restarts or when a user logs on. Policy setting changes that impact users go into effect the next time the relevant users log on. If you are using Active Directory, policy settings are updated when Active Directory re-evaluates policies at regular 90 minute intervals and applied either when the virtual desktop restarts or when a user logs on.

For some policy settings, you can enter a value or you can choose a value from a list when you add the setting to a policy. You can limit configuration of the setting by selecting Use default value. Selecting this option disables configuration of the setting and allows only the setting's default value to be used when the policy is applied. This occurs regardless of the value that was entered before selecting Use default value.

Default values for all Citrix policy settings are located in the Policy Settings Reference.

Citrix recommends the following when configuring policy settings:
- Assign policies to groups rather than individual users. If you assign policies to groups, assignments are updated automatically when you add or remove users from the group.
- Do not enable conflicting or overlapping settings in Remote Desktop Session Host Configuration. In some cases, Remote Desktop Session Host Configuration provides similar functionality to Citrix policy settings. When possible, keep all settings consistent (enabled or disabled) for ease of troubleshooting.
- Disable unused policies. Policies with no settings added create unnecessary processing.
Compare policies and templates

Jun 17, 2013

In some cases, you might need to compare the settings in a policy or template with those in other policies or templates. For example, you might need to verify setting values to ensure compliance with best practices for your environment.

You may also want to compare settings in a policy or template with the default settings provided by Citrix.

1. From Studio, select the Policy node in the left pane.
2. Click the Comparison tab and then click Select. The Select templates and policies to compare dialog box appears.
3. Select the policies or templates you want to compare. To include default values in the comparison, select the Compare to default settings checkbox.
4. Click Compare. The configured settings for the selected items are displayed in columns. Default values, if selected, are displayed in the second column by default.
5. To view all settings, select Show All Settings. To return to the default view, showing common settings across templates and policies included in the comparison, select Show Common Settings.
Apply policies

Jun 17, 2013

When you assign a policy to certain user and machine objects, that policy is applied to connections according to specific criteria or rules. If no assignments are added, the policy is applied to all connections.

In general, you can add as many assignments as you want to a policy, based on a combination of criteria.

Note: You can add only one Citrix CloudBridge assignment to a policy.

The following table lists the available assignments:

<table>
<thead>
<tr>
<th>Assignment Name</th>
<th>Assignment Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Control</td>
<td>Applies a policy based on the access control conditions through which a client is connecting.</td>
</tr>
<tr>
<td>Citrix CloudBridge</td>
<td>Applies a policy based on whether or not a user session is launched through Citrix CloudBridge.</td>
</tr>
<tr>
<td>Client IP Address</td>
<td>Applies a policy based on the IP address (IPv4 or IPv6) of the user device used to connect to the session.</td>
</tr>
<tr>
<td></td>
<td>IPv4 Examples:</td>
</tr>
<tr>
<td></td>
<td>- 12.0.0.0</td>
</tr>
<tr>
<td></td>
<td>- 12.0.0.*</td>
</tr>
<tr>
<td></td>
<td>- 12.0.0.1-12.0.0.70</td>
</tr>
<tr>
<td></td>
<td>- 12.0.0.1/24</td>
</tr>
<tr>
<td></td>
<td>IPv6 Examples:</td>
</tr>
<tr>
<td></td>
<td>- 2001:0db8:3c4d:0015:0:0:abcd:ef12</td>
</tr>
<tr>
<td></td>
<td>- 2001:0db8:3c4d:0015::/54</td>
</tr>
<tr>
<td>Client Name</td>
<td>Applies a policy based on the name of the user device from which the session is connected.</td>
</tr>
<tr>
<td>Delivery Group</td>
<td>Applies a policy based on the Delivery Group membership of the desktop running the session.</td>
</tr>
<tr>
<td>Desktop Type</td>
<td>Applies a policy based on the type of desktop running the session.</td>
</tr>
<tr>
<td>Organizational Unit</td>
<td>Applies a policy based on the organizational unit (OU) of the desktop running the session.</td>
</tr>
<tr>
<td>Tag</td>
<td>Applies a policy based on any tags applying to the desktop running the session.</td>
</tr>
<tr>
<td>User or Group</td>
<td>Applies a policy based on the user or group membership of the user connecting to the session.</td>
</tr>
</tbody>
</table>
When a user logs on, all policies that match the assignments for the connection are identified. The identified policies are sorted into priority order and multiple instances of any setting are compared. Each setting is applied according to the priority ranking of the policy. If you are using Active Directory, policy settings are updated when Active Directory re-evaluates policies at regular 90 minute intervals and applied when a user logs on.

Any policy setting that is disabled takes precedence over a lower-ranked setting that is enabled. Policy settings that are not configured are ignored.

Important: When configuring both Active Directory and Citrix policies using the Group Policy Management Console, assignments and settings may not be applied as expected. For more information, see http://support.citrix.com/article/CTX127461

By default, you are provided with an "Unfiltered" policy. The settings added to this policy apply to all connections.

If you use Studio to manage Citrix policies, settings you add to the Unfiltered policy are applied to all servers, desktops, and connections in a site.

If you have Active Directory in your environment and use the Group Policy Editor to manage Citrix policies, settings you add to the Unfiltered policy are applied to all sites and connections that are within the scope of the Group Policy Objects (GPOs) that contain the policy. For example, the Sales OU contains a GPO called Sales-US that includes all members of the US sales team. The Sales-US GPO is configured with an Unfiltered policy that includes several user policy settings. When the US Sales manager logs on to the site, the settings in the Unfiltered policy are automatically applied to the session because the user is a member of the Sales-US GPO.

An assignment's mode determines whether or not the policy is applied only to connections that match all the assignment criteria. If the mode is set to Allow (the default), the policy is applied only to connections that match the assignment criteria. If the mode is set to Deny, the policy is applied if the connection does not match the assignment criteria. The following examples illustrate how assignment modes affect Citrix policies when multiple assignments are present.

In policies with two assignments of the same type, one set to Allow and one set to Deny, the assignment set to Deny takes precedence, provided the connection satisfies both assignments. For example:

Policy 1 includes the following assignments:
- Assignment A is a User assignment that specifies the Sales group and the mode is set to Allow
- Assignment B is a User assignment that specifies the Sales manager's account and the mode is set to Deny

Because the mode for Assignment B is set to Deny, the policy is not applied when the Sales manager logs on to the site, even though the user is a member of the Sales group.

In policies with two or more assignments of differing types, set to Allow, the connection must satisfy at least one assignment of each type in order for the policy to be applied. For example:

Policy 2 includes the following assignments:
- Assignment C is a User assignment that specifies the Sales group and the mode is set to Allow
- Assignment D is a Client IP Address assignment that specifies 10.8.169.* (the corporate network) and the mode is set to
Allow

When the Sales manager logs on to the site from the office, the policy is applied because the connection satisfies both assignments.

Policy 3 includes the following assignments:
- Assignment E is a User assignment that specifies the Sales group and the mode is set to Allow
- Assignment F is an Access Control assignment that specifies NetScaler Gateway connection conditions and the mode is set to Allow

When the Sales manager logs on to the site from the office, the policy is not applied because the connection does not satisfy Assignment F.
To apply a policy

Jun 17, 2013

To apply a policy only to certain user and machines objects you must add at least one assignment to that policy. If you do not add any assignments, policy settings are applied to all connections, unless those policy settings are overridden by settings in a policy with a higher priority.

1. From Studio, select the Policy node in the left pane.
2. Click the Policies tab and then select an existing policy or create a new policy.
3. Follow the policy wizard to the Users and Machines page.
4. Select the assignment you want to apply and click Assign.
5. In the Assign Policy dialog box, select the mode for the assignment and configure the assignment elements:

<table>
<thead>
<tr>
<th>Assignment type</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Control</td>
<td>Connection type. Select whether to apply the policy to connections made either with or without NetScaler Gateway. Access Control. Enter the name of the NetScaler Gateway (formerly Access Gateway) virtual server. Access condition. Enter the name of the endpoint analysis policy or session policy to use.</td>
</tr>
<tr>
<td>Citrix CloudBridge</td>
<td>Connections. Select whether to apply the policy to sessions launched either with or without Citrix CloudBridge.</td>
</tr>
<tr>
<td>Client IP address</td>
<td>IP address. IP address of the user device to which you want to apply the policy.</td>
</tr>
<tr>
<td>Client name</td>
<td>Client name. Name of the user device to which you want to apply the policy.</td>
</tr>
<tr>
<td>Delivery Group</td>
<td>Controller. Name of the controller managing the desktops to which you want to apply the policy. Delivery Group. Name of the Delivery Group to which you want to apply the policy.</td>
</tr>
<tr>
<td>Desktop type</td>
<td>Desktop type. Type of desktop to which you want to apply the policy. Choose one of the following: Private Desktop, Shared Desktop, Private Application, Shared Application</td>
</tr>
<tr>
<td>Organizational Unit (OU)</td>
<td>Organizational Unit. Name of the OU to which to apply the policy.</td>
</tr>
<tr>
<td>Tag</td>
<td>Controller. Name of the controller managing the desktops to which you want to apply the policy. Tag. The desktop tag to search for when applying the policy.</td>
</tr>
<tr>
<td>Assignment type</td>
<td>Parameters</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------</td>
</tr>
<tr>
<td>User or group</td>
<td>User or group name. Name of the user or group to which you want to apply the policy.</td>
</tr>
</tbody>
</table>

The policy is applied the next time the relevant users establish a connection.
Customize your environment with multiple policies

Mar 21, 2015

You can use multiple policies to customize your environment to meet users’ needs based on their job functions, geographic locations, or connection types. For example, for security reasons you may need to place restrictions on user groups who regularly work with highly sensitive data. You can create a policy that prevents users from saving sensitive files on their local client drives. However, if some people in the user group do need access to their local drives, you can create another policy for only those users. You then rank or prioritize the two policies to control which one takes precedence.

When using multiple policies, you need to determine how to prioritize them, how to create exceptions, and how to view the effective policy when policies conflict.

In general, policies override similar settings configured for the entire site, for specific controllers, or on the user device. The exception to this principle is security. The highest encryption setting in your environment, including the operating system and the most restrictive shadowing setting, always overrides other settings and policies.

Citrix policies interact with policies you set in your operating system. In a Citrix environment, Citrix settings override the same settings configured in an Active Directory policy or using Remote Desktop Session Host Configuration. This includes settings that are related to typical Remote Desktop Protocol (RDP) client connection settings such as Desktop wallpaper, Menu animation, and View window contents while dragging. For some policy settings, such as Secure ICA, the settings in policies must match the settings in the operating system. If a higher priority encryption level is set elsewhere, the Secure ICA policy settings that you specify in the policy or when you are delivering application and desktops can be overridden.

For example, the encryption settings that you specify when you are creating delivery groups to provide users with applications and desktops should be at the same level as the encryption settings you specified throughout your environment.

Prioritizing policies allows you to define the precedence of policies when they contain conflicting settings. The process used to evaluate policies is as follows:

1. When a user logs on, all policies that match the assignments for the connection are identified.
2. The identified policies are sorted into priority order and multiple instances of any setting are compared. Each setting is applied according to the priority ranking of the policy.

You prioritize policies by giving them different priority numbers. By default, new policies are given the lowest priority. If policy settings conflict, a policy with a higher priority (a priority number of 1 is the highest) overrides a policy with a lower priority. Settings are merged according to priority and the setting’s condition; for example, whether the setting is disabled or enabled. Any disabled setting overrides a lower-ranked setting that is enabled. Policy settings that are not configured are ignored and do not override the settings of lower-ranked settings.

When you create policies for groups of users, user devices, or machines, you may find that some members of the group require exceptions to some policy settings. You can create exceptions by:

- Creating a policy only for those group members who need the exceptions and then ranking the policy higher than the policy for the entire group
- Using the Deny mode for an assignment added to the policy

An assignment with the mode set to Deny applies a policy only to connections that do not match the assignment criteria. For example, a policy contains the following assignments:

- Assignment A is a Client IP address assignment that specifies the range 208.77.88.* and the mode is set to Allow
Assignment B is a User assignment that specifies a particular user account and the mode is set to Deny.

The policy is applied to all users who log on to the site with IP addresses in the range specified in Assignment A. However, the policy is not applied to the user logging on to the site with the user account specified in Assignment B, even though the user's computer is assigned an IP address in the range specified in Assignment A.
Determine which policies apply to a connection

Jun 28, 2013

Sometimes a connection does not respond as expected because multiple policies apply. If a higher priority policy also applies to a connection, it can override the settings you configure in the original policy. You can determine how final policy settings are merged for a connection by calculating the Resultant Set of Policy.

You can calculate the Resultant Set of Policy in the following ways:

- Use the Citrix Group Policy Modeling Wizard to simulate a connection scenario and discern how Citrix policies might be applied
- Use Group Policy Results to produce a report describing the Citrix policies in effect for a given user and controller

You can launch the Citrix Group Policy Modeling Wizard from the Action pane in Studio. If your environment includes Active Directory, you can launch both tools from the Group Policy Management Console in Windows.

With the Citrix Group Policy Modeling Wizard, you can specify conditions for a connection scenario such as domain controller, users, Citrix policy assignment evidence values, and simulated environment settings such as slow network connection. The report that the wizard produces lists the Citrix policies that would likely take effect in the scenario.

If you are logged on to the controller as a domain user and your environment includes Active Directory, the wizard calculates the Resultant Set of Policy using both site policy settings and Active Directory Group Policy Objects (GPOs).

The Group Policy Results tool helps you evaluate the current state of GPOs in your environment and generates a report that describes how these objects, including Citrix policies, are currently being applied to a particular user and controller.

If you run the Citrix Group Policy Modeling Wizard or Group Policy Results tool from the Group Policy Management Console, site policy settings created using Studio are not included in the Resultant Set of Policy.

To ensure you obtain the most comprehensive Resultant Set of Policy, Citrix recommends launching the Citrix Group Policy Modeling wizard from Studio, unless you create policies using only the Group Policy Management Console.

Users, IP addresses, and other assigned objects can have multiple policies that apply simultaneously. This can result in conflicts where a policy may not behave as expected. When you run the Citrix Group Policy Modeling Wizard or the Group Policy Results tool, you might discover that no policies are applied to user connections. When this happens, users connecting to their applications and desktops under conditions that match the policy evaluation criteria are not affected by any policy settings. This occurs when:

- No policies have assignments that match the policy evaluation criteria
- Policies that match the assignment do not have any settings configured
- Policies that match the assignment are disabled

If you want to apply policy settings to the connections that meet the specified criteria:

- Make sure the policies that you want to apply to those connections are enabled
• Make sure the policies that you want to apply have the appropriate settings configured
To simulate connection scenarios with Citrix policies

Mar 05, 2013

1. Depending on your environment, open the Citrix Group Policy Modeling Wizard:
   - From Studio, select the Policy node in the left pane of the console and then click the Modeling tab. From the Actions pane, select Launch Modeling Wizard.
   - From the Group Policy Management Console, right-click the Citrix Group Policy Modeling node in the tree pane and then select Citrix Group Policy Modeling Wizard.

2. Follow the instructions in the wizard to select the domain controller, users, computers, environment settings, and Citrix assignment criteria you want to use in the simulation.

When you click Finish, the wizard produces a report of the modeling results.

In Studio, the report appears in the middle pane under the Modeling tab.

Manage policies using Group Policy Editor

Mar 28, 2015

If your network environment includes Active Directory and you have the appropriate permissions to manage Group Policy, you can use the Group Policy Editor to create policies for your site. The settings you configure affect the GPOs you specify through the Group Policy Management Console.

Important: You must use Group Policy Editor to configure certain settings, for example, settings relating to registering virtual desktops with a controller.

1. From Group Policy Editor, expand the Computer Configuration or User Configuration nodes, expand the Policies node, and then select Citrix Policies.
2. Do one of the following:

<table>
<thead>
<tr>
<th>Task</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>To create a new policy</td>
<td>On the Policies tab, click New.</td>
</tr>
<tr>
<td>To edit an existing policy</td>
<td>On the Policies tab, select the policy you want to edit and then click Edit.</td>
</tr>
<tr>
<td>To change the priority of an existing policy</td>
<td>On the Policies tab, select the policy for which you want to change the priority and then click either Higher or Lower.</td>
</tr>
<tr>
<td>To view summary information about a policy</td>
<td>On the Policies tab, select the policy for which you want to view summary information you want to view and then click the Summary tab.</td>
</tr>
<tr>
<td>To view and amend settings configured for a policy</td>
<td>On the Policies tab, select the policy for which you want to view or amend settings then click the Settings tab.</td>
</tr>
<tr>
<td>To view and amend filters configured for a policy</td>
<td>On the Policies tab, select the policy for which you want to view or amend filters and then click the Filters tab.</td>
</tr>
<tr>
<td>To enable or disable a policy</td>
<td>On the Policies tab, select the policy you want to enable or disable, and then select either Actions &gt; Enable or Actions &gt; Disable.</td>
</tr>
<tr>
<td>To create a new template from an existing policy</td>
<td>On the Policies tab, select the policy from which you want to create a new template, and then select Actions &gt; Save as Template.</td>
</tr>
<tr>
<td>To create a new policy from an existing template</td>
<td>On the Templates tab, select the template from which you want to create a new policy, and then click New Policy.</td>
</tr>
<tr>
<td>Task</td>
<td>Instructions</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>To create a new template from an existing template</td>
<td>On the Templates tab, select the template from which you want to create a new template, and then click New Template.</td>
</tr>
<tr>
<td>To import a template</td>
<td>On the Templates tab, select Actions &gt; Import.</td>
</tr>
<tr>
<td>To export a template</td>
<td>On the Templates tab, select Actions &gt; Export.</td>
</tr>
<tr>
<td>To view template settings</td>
<td>On the Template tab, select the template for which you want to view settings and then click the Settings tab.</td>
</tr>
<tr>
<td>To view a summary of template properties</td>
<td>On the Template tab, select the template for which you want to view summary properties and then click the Properties tab.</td>
</tr>
<tr>
<td>To view any template prerequisites</td>
<td>On the Template tab, select the template for which you want to view any prerequisites and then click the Prerequisites tab.</td>
</tr>
</tbody>
</table>
User profiles

Jun 28, 2013

By default, Citrix Profile management 5.1 is installed silently on master images when you install the Virtual Delivery Agent but you do not have to use Profile management as a profile solution.

To suit your users' varying needs, using XenDesktop policies you can apply different profile behavior to the machines in each Delivery Group. For example, one Delivery Group might require Citrix mandatory profiles, whose template is stored in one network location, but another Delivery Group might require Citrix roaming profiles stored in another location with several redirected folders.

Whichever profile solution you choose, Director administrators can access diagnostic information for and troubleshoot user profiles. For more information, see the Director documentation.

If other administrators in your organization are responsible for XenDesktop policies, work with them to ensure that they set any profile-related policies across your Delivery Groups.

Be aware that Profile management policies can also be set in Group Policy, in the Profile management .ini file, and locally on individual virtual machines. These multiple ways of defining profile behavior are read in the following order:

1. Group Policy (.adm or .admx files)
2. XenDesktop policies in the Policy node
3. Local policies on the virtual machine that the user connects to
4. Profile management .ini file

For example, if you configure the same policy in both Group Policy and the Policy node, the system reads the policy setting in Group Policy and ignores the XenDesktop policy setting.

If you use the Personal vDisk feature of XenDesktop, Citrix user profiles are stored on virtual desktops' Personal vDisks by default. Do not delete the copy of a profile in the user store while a copy remains on the Personal vDisk. Doing so creates a Profile management error, and causes a temporary profile to be used for logons to the virtual desktop. For more information on this, see the topic — Users Receive New or Temporary Profiles in the Profile management documentation.
How automatic configuration works

Jun 20, 2013

XenDesktop automatically detects the desktop type based on the Virtual Delivery Agent installation and, in addition to the configuration choices you make in Studio, sets Profile management defaults accordingly.

The policies that Profile management adjusts when it detects a XenDesktop deployment are shown in the table. Any non-default policy settings are preserved and are not overwritten by this feature. Consult the Profile management documentation for information on each policy.

The types of machines that create profiles affect the policies that are adjusted. The primary factors are whether machines are persistent or provisioned, and whether they are shared by multiple users or dedicated to just one user.

Persistent systems have some type of local storage, the contents of which can be expected to persist when the system turns off. Persistent systems may employ storage technology such as storage area networks (SANs) to provide local disk mimicking. In contrast, provisioned systems are created “on the fly” from a base disk and some type of identity disk. Local storage is usually mimicked by a RAM disk or network disk, the latter often provided by a SAN with a highspeed link. The provisioning technology is generally Provisioning Services or Machine Creation Services (or a third-party equivalent). Sometimes provisioned systems have persistent local storage, which may be provided by Personal vDisks; these are classed as persistent.

Together, these two factors define the following machine types:

- **Both persistent and dedicated** -- Examples are Desktop OS machines with a static assignment and a Personal vDisk that are created with Machine Creation Services (in XenDesktop), desktops with Personal vDisks that are created with VDI-in-a-Box, physical workstations, and laptops
- **Both persistent and shared** -- Examples are Server OS machines that are created with Machine Creation Services (in XenDesktop), and XenApp servers
- **Both provisioned and dedicated** -- Examples are Desktop OS machines with a static assignment but without a Personal vDisk that are created with Provisioning Services (in XenDesktop)
- **Both provisioned and shared** -- Examples are Desktop OS machines with a random assignment that are created with Provisioning Services (in XenDesktop), desktops without Personal vDisks that are created with VDI-in-a-Box, and XenApp servers

The following Profile management policy settings are suggested guidelines for the different machine types. They work well in most cases, but you may want to deviate from these as your deployment requires.

**Important:** Delete locally cached profiles on logoff, Profile streaming, and Always cache are enforced by the auto-configuration feature. Adjust the other policies manually.

<table>
<thead>
<tr>
<th>Policy</th>
<th>Both persistent and dedicated</th>
<th>Both persistent and shared</th>
<th>Both provisioned and dedicated</th>
<th>Both provisioned and shared</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete locally cached profiles on logoff</td>
<td>Disabled</td>
<td>Enabled</td>
<td>Disabled (note 5)</td>
<td>Enabled</td>
</tr>
<tr>
<td>Profile streaming</td>
<td>Disabled</td>
<td>Enabled</td>
<td>Enabled</td>
<td>Enabled</td>
</tr>
<tr>
<td>Policy</td>
<td>Both persistent and dedicated</td>
<td>Both persistent and shared</td>
<td>Both provisioned and dedicated</td>
<td>Both provisioned and shared</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------------------</td>
<td>----------------------------</td>
<td>--------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>Always cache</td>
<td>Disabled</td>
<td>Disabled</td>
<td>Enabled</td>
<td>Enabled</td>
</tr>
<tr>
<td>Active write back</td>
<td>Disabled</td>
<td>Disabled (note 3)</td>
<td>Enabled</td>
<td>Enabled</td>
</tr>
<tr>
<td>Process logons of local administrators</td>
<td>Enabled</td>
<td>Disabled (note 4)</td>
<td>Enabled</td>
<td>Enabled (note 7)</td>
</tr>
</tbody>
</table>

**Notes**

1. Because Profile streaming is disabled for this machine type, the Always cache setting is always ignored.
2. Disable Always cache. However, you can ensure that large files are loaded into profiles as soon as possible after logon by enabling this policy and using it to define a file size limit (in MB). Any file this size or larger is cached locally as soon as possible.
3. Disable Active write back except to save changes in profiles of users who roam between XenApp servers. In this case, enable this policy.
4. Disable Process logons of local administrators except for Hosted Shared Desktops. In this case, enable this policy.
5. Disable Delete locally cached profiles on logoff. This retains locally cached profiles. Because the machines are reset at logoff but are assigned to individual users, logons are faster if their profiles are cached.
6. Disable Always cache. However, you can ensure that large files are loaded into profiles as soon as possible after logon by enabling this policy and using it to define a file size limit (in MB). Any file this size or larger is cached locally as soon as possible.
7. Enable Process logons of local administrators except for profiles of users who roam between XenApp servers. In this case, disable this policy.
Folder redirection

May 03, 2013
You can configure folder redirection using XenDesktop policies in Studio. Folder redirection lets you store user data on network shares other than the location where the profiles are stored. This reduces profile size and load time but it might impact network bandwidth.

Folder redirection does not require that Citrix user profiles are employed. You can choose to manage user profiles on your own, and still redirect folders.

General advice on folder redirection

When setting up folder redirection in Studio, note the following:

- Ensure that the network locations used to store the contents of redirected folders are available and have the correct permissions. XenDesktop validates these properties of the locations for you.
- Redirected folders are set up on the network and their contents populated from users' virtual desktops at logon.

Advanced folder redirection

In deployments with multiple operating systems (OSs), you might want some of a user's profile to be shared by each OS. The rest of the profile is not shared and is used only by one OS. To ensure a consistent user experience across the OSs, you need a different configuration for each OS. This is advanced folder redirection. For example, different versions of an application running on two OSs might need to read or edit a shared file, so you decide to redirect it to a single network location where both versions can access it. Alternatively, because the Start Menu folder contents are structured differently in two OSs, you decide to redirect only one folder, not both. This separates the Start Menu folder and its contents on each OS, ensuring a consistent experience for users.

If your deployment requires advanced folder redirection, you must understand the structure of your users' profile data and determine which parts of it can be shared between OSs. This is important because unpredictable behavior can result unless folder redirection is used correctly.

To redirect folders in advanced deployments

1. Use a separate Delivery Group for each OS.
2. Understand where your virtual applications, including those on virtual desktops, store user data and settings, and understand how the data is structured.
3. For shared profile data that can safely roam (because it is structured identically in each OS), redirect the containing folders in each Delivery Group.
4. For non-shared profile data that cannot roam, redirect the containing folder in only one of the Desktop Groups, typically the one with the most used OS or the one where the data is most relevant. Alternatively, for non-shared data that cannot roam between OSs, redirect the containing folders on both systems to separate network locations.

Example advanced deployment

This example has applications, including versions of Microsoft Outlook and Internet Explorer, running on Windows 8 desktops and applications, including other versions of Outlook and Internet Explorer, delivered by Windows Server 2008. To achieve this, you have already set up two Delivery Groups for the two OSs. Users want to access the same set of Contacts and Favorites in both versions of those two applications.
Important: The following decisions and advice are valid for the OSs and deployment described. In your organization, the folders you choose to redirect and whether you decide to share them depend on a number of factors that are unique to your specific deployment.

Using XenDesktop policies applied to the Delivery Groups, you choose the following folders to redirect.

<table>
<thead>
<tr>
<th>Folder</th>
<th>Redirected in Windows 8?</th>
<th>Redirected in Windows Server 2008?</th>
</tr>
</thead>
<tbody>
<tr>
<td>My Documents</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Application Data</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Contacts</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Desktop</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Downloads</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Favorites</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Links</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>My Music</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>My Pictures</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>My Videos</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Searches</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Saved Games</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Start Menu</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Note the following about the shared, redirected folders:

- After analyzing the structure of the data saved by the different versions of Outlook and Internet Explorer, you decide it is safe to share the Contacts and Favorites folders.
- You know the structure of the My Documents, My Music, My Pictures, and My Videos folders is standard across OSs, so it is safe to store these in the same network location for each Delivery Group.

Note the following about the non-shared, redirected folders:

- You do not redirect the Desktop, Links, Searches, or Start Menu folders folder in the Windows Server Delivery Group because data in these folders is organized differently in the two OSs. It therefore cannot be shared.
- To ensure predictable behavior of this non-shared data, you redirect it only in the Windows 8 Delivery Group. You choose this, rather than the Windows Server Delivery Group, because Windows 8 will be used more often by users in their day-to-day work; they will only occasionally access the applications delivered by the server. Also, in this case the non-shared data is more relevant to a desktop environment rather than an application environment. For example, desktop shortcuts are stored in the Desktop folder and might be useful if they originate from a Windows 8 machine but not from a Windows Server machine.
Note the following about the non-redirected folders:

- You do not want to clutter your servers with users' downloaded files, so you choose not to redirect the Downloads folder
- Data from individual applications can cause compatibility and performance issues, so you decide not to redirect the Application Data folder


Folder redirection and exclusions

In Citrix Profile management (but not in Studio), a performance enhancement allows you to prevent folders from being processed using exclusions. If you use this feature, do not exclude any redirected folders. The folder redirection and exclusion features work together, so ensuring no redirected folders are excluded allows Profile management to move them back into the profile folder structure again, while preserving data integrity, if you later decide not to redirect them.
Maintain session activity

Apr 09, 2013

Maintaining session activity is important to ensure you provide the best user experience. Losing connectivity due to unreliable networks, highly variable network latency, and range limitations of wireless devices, can lead to user frustration. Being able to move quickly between workstations, accessing the same set of applications each time they log on, is a priority for many mobile workers, such as health-care workers in a hospital.

Use the following features to optimize the reliability of sessions, reduce the amount of inconvenience, downtime, and loss of productivity users incur due to lost network connectivity, and provide mobile users with the ability to roam quickly and easily between devices.

- Session Reliability
- Auto Client Reconnect
- ICA Keep-Alive
- Workspace control
Configure Session Reliability

Apr 30, 2013

Session Reliability keeps sessions active and on the user's screen when network connectivity is interrupted. Users continue to see the application they are using until network connectivity resumes.

This feature is especially useful for mobile users with wireless connections. Take, for example, a user with a wireless connection who enters a railroad tunnel and momentarily loses connectivity. Ordinarily, the session is disconnected and disappears from the user's screen, and the user has to reconnect to the disconnected session.

With Session Reliability, the session remains active on the server. To indicate that connectivity is lost, the user's display freezes and the cursor changes to a spinning hourglass until connectivity resumes on the other side of the tunnel. The user continues to access the display during the interruption and can resume interacting with the application when the network connection is restored. Session Reliability reconnects users without reauthentication prompts.

Citrix Receiver users cannot override the server setting.

You can use Session Reliability with Secure Sockets Layer (SSL).

Note: SSL encrypts only the data sent between the user device and NetScaler Gateway.

By default, Session Reliability is enabled through policy settings. You can edit the port on which the server listens for session reliability traffic and edit the amount of time Session Reliability keeps an interrupted session connected.

The Session reliability connections policy setting allows or prevents session reliability.

The Session reliability timeout policy setting has a default of 180 seconds, or three minutes. Though you can extend the amount of time Session Reliability keeps a session open, this feature is designed to be convenient to the user and it does not, therefore, prompt the user for reauthentication. As you extend the amount of time a session is kept open, chances increase that a user may get distracted and walk away from the user device, potentially leaving the session accessible to unauthorized users.

Incoming session reliability connections use port 2598, unless you change the port number defined in the Session reliability port number policy setting.

If you do not want users to be able to reconnect to interrupted sessions without having to reauthenticate, use the Auto Client Reconnect feature. You can configure the Auto client reconnect authentication policy setting to prompt users to reauthenticate when reconnecting to interrupted sessions.

If you use both Session Reliability and Auto Client Reconnect, the two features work in sequence. Session Reliability closes, or disconnects, the user session after the amount of time you specify in the Session reliability timeout policy setting. After that, the Auto Client Reconnect policy settings take effect, attempting to reconnect the user to the disconnected session.
Configure Automatic Client Reconnection

Apr 22, 2013

Users can be disconnected from their sessions because of unreliable networks, highly variable network latency, or range limitations of wireless devices. With the Auto Client Reconnect feature, Receiver can detect unintended disconnections of ICA sessions and reconnect users to the affected sessions automatically. When this feature is enabled on the server, users do not have to reconnect manually to continue working. Receiver attempts to reconnect to the session until there is a successful reconnection or the user cancels the reconnection attempts.

Configure Auto Client Reconnect using the following policy settings:

- Auto client reconnect. Enables or disables automatic reconnection by Receiver after a connection has been interrupted.
- Auto client reconnect authentication. Enables or disables the requirement for user authentication upon automatic reconnection.
- Auto client reconnect logging. Enables or disables logging of reconnection events in the event log. Logging is disabled by default. When enabled, the server's System log captures information about successful and failed automatic reconnection events. Each server stores information about reconnection events in its own System log; the server farm does not provide a combined log of reconnection events for all servers.

Auto Client Reconnect incorporates an authentication mechanism based on encrypted user credentials. When a user initially logs on to a site, the server encrypts and stores the user credentials in memory, and creates and sends a cookie containing the encryption key to Receiver. Receiver submits the key to the server for reconnection. The server decrypts the credentials and submits them to Windows logon for authentication. When cookies expire, users must reauthenticate to reconnect to sessions.

Cookies are not used if you enable the Auto client reconnection authentication setting. Instead, users are presented with a dialog box to users requesting credentials when Receiver attempts to reconnect automatically.

Note: For maximum protection of users' credentials and sessions, use SSL encryption for all communication between clients and the server farm.

Disable Auto Client Reconnect on Citrix Receiver for Windows by using the icaclient.adm file. For more information, see the — Receiver for Windows documentation, in eDocs.

Settings for connections also affect Auto Client Reconnect.

Configure connections for Automatic Client Reconnection

By default, Auto Client Reconnect is enabled through policy settings at site level. User reauthentication is not required. However, if a server's ICA TCP connection is configured to reset sessions with a broken communication link, automatic reconnection does not occur. Auto Client Reconnect works only if the server disconnects sessions when there is a broken or timed out connection.

In this context, the ICA TCP connection refers to a server's virtual port (rather than an actual network connection) that is used for sessions on TCP/IP networks.

By default, the ICA TCP connection on a server is set to disconnect sessions with broken or timed out connections. Disconnected sessions remain intact in system memory and are available for reconnection by Receiver.

The connection can be configured to reset, or log off, sessions with broken or timed out connections. When a session is...
reset, attempting to reconnect initiates a new session; rather than restoring a user to the same place in the application in use, the application is restarted.

If the server is configured to reset sessions, Auto Client Reconnect creates a new session. This process requires users to enter their credentials to log on to the server.

Automatic reconnection can fail if Receiver or the plug-in submits incorrect authentication information, which might occur during an attack or the server determines that too much time has elapsed since it detected the broken connection.
Configure ICA Keep-Alive

Mar 26, 2013

Enabling the ICA Keep-Alive feature prevents broken connections from being disconnected. When enabled, if the server detects no activity (for example, no clock change, no mouse movement, no screen updates), this feature prevents Remote Desktop Services from disconnecting that session. The server sends keep-alive packets every few seconds to detect if the session is active. If the session is no longer active, the server marks the session as disconnected.

Note: The ICA Keep-Alive feature works only if you are not using Session Reliability. Session Reliability has its own mechanisms to prevent broken connections from being disconnected. Configure ICA Keep-Alive only for connections that do not use Session Reliability. ICA Keep-Alive settings override keep-alive settings that are configured in Microsoft Windows Group Policy.

Configure ICA Keep-Alive using the following policy settings:

- **ICA keep alive timeout.** Specifies the interval (1-3600 seconds) used to send ICA keep-alive messages. Do not configure this option if you want your network monitoring software to close inactive connections in environments where broken connections are so infrequent that allowing users to reconnect to sessions is not a concern. The 60 second default interval causes ICA Keep-Alive packets to be sent to user devices every 60 seconds. If a user device does not respond in 60 seconds, the status of the ICA sessions changes to disconnected.

- **ICA keep alives.** Sends or prevents sending ICA keep-alive messages periodically.
Configure workspace control

May 01, 2014

Workspace control lets desktops and applications follow a user from one device to another. This ability to roam enables a user to access all desktops or open applications from anywhere simply by logging on, without having to restart the desktops or applications on each device. For example, workspace control can assist health-care workers in a hospital who need to move quickly among different workstations and access the same set of applications each time they log on. If you configure workspace control options to allow it, these workers can disconnect from multiple applications at one client device and then reconnect to open the same applications at a different client device.

Workspace control affects the following activities:

- **Logging on** – By default, workspace control enables users to reconnect automatically to all running desktops and applications when logging on, bypassing the need to reopen them manually. Through workspace control, users can open disconnected desktops or applications, as well as any that are active on another client device. Disconnecting from a desktop or application leaves it running on the server. If you have roaming users who need to keep some desktops or applications running on one client device while they reconnect to a subset of their desktops or applications on another client device, you can configure the logon reconnection behavior to open only the desktops or applications that the user disconnected from previously.

- **Reconnecting** – After logging on to the server, users can reconnect to all of their desktops or applications at any time by clicking Reconnect. By default, Reconnect opens desktops or applications that are disconnected, plus any that are currently running on another client device. You can configure Reconnect to open only those desktops or applications that the user disconnected from previously.

- **Logging off** – For users opening desktops or applications through StoreFront, you can configure the Log Off command to log the user off from StoreFront and all active sessions together, or log off from StoreFront only.

- **Disconnecting** – Users can disconnect from all running desktops and applications at once, without needing to disconnect from each individually.

Workspace control is available only for Receiver users who access desktops and applications through a Citrix StoreFront connection. By default, workspace control is disabled for virtual desktop sessions, but is enabled for hosted applications. Session sharing does not occur by default between published desktops and any published applications running inside those desktops.

User policies, client drive mappings, and printer configurations change appropriately when a user moves to a new client device. Policies and mappings are applied according to the client device where the user is currently logged on to the session. For example, if a health care worker logs off from a client device in the emergency room of a hospital and then logs on to a workstation in the hospital's X-ray laboratory, the policies, printer mappings, and client drive mappings appropriate for the session in the X-ray laboratory go into effect at the session startup.

You can customize what printers appear to users when they change locations. You can also control whether users can print to local printers, how much bandwidth is consumed when users connect remotely, and other aspects of their printing experiences.

For information about enabling and configuring workspace control for users, see the StoreFront documentation in eDocs.
Personal vDisks

Jun 21, 2013

The Personal vDisk feature in XenDesktop retains the single image management of pooled and streamed desktops while allowing people to install applications and change their desktop settings.

Unlike traditional Virtual Desktop Infrastructure (VDI) deployments involving pooled desktops, where users lose their customizations and personal applications when the administrator alters the base virtual machine (VM), deployments using Personal vDisks retain those changes. This means administrators can easily and centrally manage their base VMs while providing users with a customized and personalized desktop experience.

Personal vDisks provide this separation by redirecting all changes made on the user's VM to a separate disk (the Personal vDisk) attached to the user's VM. The content of the Personal vDisk is blended at runtime with the content from the base VM to provide a unified experience. In this way, users can still access applications provisioned by their administrator in the base VM.

Personal vDisks have two parts, which are by default equally sized:

- One part comprises C:\Users (in Windows 7) or C:\Documents and Settings (in Windows XP). This contains user data, documents, and the user profile. By default this uses drive P: but you can choose a different drive letter when you use Studio to create a machine catalog with Personal vDisks.
- The other part comprises a Virtual Hard Disk file (a .vhd file). This contains all other items, for example applications installed in C:\Program Files. This part is hidden from users; it is not displayed in Windows Explorer.

Personal vDisks support the provisioning of department-level applications, as well as applications downloaded and installed by users, including those that require drivers, databases, and PC management software. If a user's change conflicts with an administrator's change, a Personal vDisk provides a simple and automatic way to reconcile the changes.

In addition, locally administered applications (such as those provisioned and managed by local IT departments) can also be provisioned into the user's environment. The user experiences no difference in usability; Personal vDisks ensure all changes made and all applications installed are stored on the vDisk. Where an application on a Personal vDisk exactly matches one on a master image, the copy on the Personal vDisk is discarded to save space without the user losing access to the application.

An algorithm automatically adjusts the relative sizes of the two parts of the vDisk depending on how the vDisk is used. For example, if a user installs several big applications on the Personal vDisk so space becomes limited, the application space is increased relative to the space for user data. The overall size of the Personal vDisk does not change. You can configure this resizing feature.

Physically, you store Personal vDisks on the hypervisor but they do not have to be in the same location as other disks attached to the virtual desktop. This can make Personal vDisk storage cheaper.
Manage Personal vDisks

Apr 15, 2015

Storage for hosts with Personal vDisks

Note: You can upgrade this feature using Personal vDisk 7.0. This release contains new features and bug fixes, and is available as a download.

When you create a host, you define storage locations for disks that are used by virtual machines. You can separate the Personal vDisks, which store the user profiles and user-installed applications, from the disks used for the machines' operating system. Each virtual machine must have access to a storage location for both disks. If you use local storage for both, they must be accessible from the same hypervisor. To ensure this requirement is met, Studio offers you only compatible storage locations when you create the host.

Back up Personal vDisks regularly using any preferred method. The vDisks are standard volumes in a hypervisor's storage tier, so you can back them up just like any other volume.

Add Personal vDisks to existing hosts

You add Personal vDisks to new hosts when you configure a new XenDesktop site. You can also add Personal vDisks and storage for them to existing hosts (but not machine catalogs).

1. In Studio, click Configuration > Hosting and select a host.
2. Click Add personal vDisk storage and specify the storage location.

Run the inventory when updating master images

You enable the Personal vDisk feature for use with a master image when you install the Virtual Delivery Agent. During the installation procedure and after any update to the image after installation, it is important that the disk's inventory is refreshed and a new snapshot is created. This procedure describes the required steps.

Because administrators, not users, manage master images, if you install an application that places binary files in the administrator's user profile, the application is not available to users of shared virtual desktops (including those based on pooled machine catalogs and pooled with Personal vDisk machine catalogs). Users must install such applications themselves.

It is best practice to take a snapshot of the image after each step in this procedure.

1. Update the master image by installing any applications or operating system updates, and performing any system configuration on the machine. Other information on this step is provided in the topic Prepare a master image.
   For master images based on Windows XP that you plan to deploy with Personal vDisks, check that no dialog boxes are open (for example, messages confirming software installations or prompts to use unsigned drivers). Open dialog boxes on master images in this environment prevent the Virtual Delivery Agent from registering with the Delivery Controller. You can prevent prompts for unsigned drivers using the Control Panel. For example, on Windows XP click System > Hardware > Driver Signing, and select the option to ignore warnings.
2. Shut down the machine. For Windows 7 machines, click Cancel when Citrix Personal vDisk blocks the shutdown.
3. In the Citrix Personal vDisk dialog box, click Update Inventory. This step may take several minutes to complete.
   Important: If you interrupt the following shutdown (even to make a minor update to the image), the Personal vDisk's inventory no longer matches the master image. This causes the Personal vDisk feature to stop working. If you interrupt the shutdown, you must restart the machine, shut it down, and when prompted click Update Inventory again.
4. When the inventory operation shuts down the machine, take a snapshot of the master image.

Adjust the space available for applications

You can manually adjust the automatic resizing algorithm that determines the size of the VHD relative to the P: drive. Typically, there is no need to make this adjustment because XenDesktop manages the split dynamically on its own. But if, for example, you know users will install a number of applications that are too big to fit on the VHD even after it is resized by the algorithm, you could increase the initial size of the application space to accommodate the user-installed applications. You adjust the space by editing the initial size of the VHD in the registry. Preferably, you make this adjustment on a machine catalog's master image (that is, before the desktops in the machine catalog are released to users). Alternatively, you can adjust the size of the VHD on a virtual desktop, when a user finds that they do not have the space to install an application, but you must repeat this operation individually on each affected virtual desktop; you cannot adjust the size in a machine catalog that is already in use.

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

1. On the master image or desktop, locate the registry keys located in \HKEY_LOCAL_MACHINE\SOFTWARE\Citrix\personal vDisk\Config. You use several of these keys in the rest of this procedure.
   1. Set MinimumVHDSizeMB to the desired new initial size of the VHD (in megabytes). The new size must be greater than the existing size but less than the size of the physical disk minus PvDReservedSpaceMB.
   2. Ensure that PercentOfPvDForApps is set to 50. This sets the default allocation of space on the personal vDisk to 50%. If any other value is used, the dynamic resizing algorithm is disabled.
   3. Enable the algorithm by setting EnableDynamicResizeOfAppContainer to 1.
   4. If you are using a profile management solution (such as Citrix Profile management), check that EnableUserProfileRedirection is set to 0. This value ensures that all of the space on P: is allocated to applications. Important: In XenDesktop 7.1, changes to the EnableUserProfileRedirection registry key are not honored when you perform an image update. If you use this version of XenDesktop, set the key's value when you first create the catalogs from which the profiles will originate. You cannot modify the redirection behavior later.

If you are carrying out this operation on a virtual desktop rather than a machine catalog, resizing takes place when the desktop is restarted.

Disable automatic resizing

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

1. On the master image or desktop, set EnableDynamicResizeOfAppContainer to 0. This registry key is located in \HKEY_LOCAL_MACHINE\SOFTWARE\Citrix\personal vDisk\Config.

Antivirus products

If antivirus products are installed on your desktops, ensure the VHD is big enough to store antivirus definition files, which are typically large.

The presence of antivirus products can affect how long it takes to run the inventory or perform an update. Performance can improve if you add CtxPvD.exe and CtxPvDSvc.exe to the exclusion list of your antivirus product. These files are located in C:\Program Files\Citrix\personal vDisk\bin.
Change the assignment of Personal vDisks

Important: In Provisioning Services, you can change the desktop that a Personal vDisk is assigned to in a machine catalog. When you do this, you must first put the desktop into maintenance mode in Studio, make the reassignment in Provisioning Services, and then wait for the desktop to re-register with the Delivery Controller before taking it out of maintenance mode. For instructions on reassigning desktops in machine catalogs, see the Provisioning Services documentation.

Back up and restore Personal vDisks

Two PowerShell scripts supplied on the XenDesktop media allow you to back up and restore Personal vDisks. The scripts are located in the Support\Tools\Scripts folder:

- migration-backup.ps1 captures the mapping between each user and their Personal vDisk in a machine catalog and stores this information in an .xml file.
- migration-restore.ps1 uses the .xml file to re-create a user's desktop in a machine catalog.

Important: Before backing up and restoring, note the following:

- The scripts work with the hypervisor API so the hypervisor's PowerShell snap-in must be installed on the Delivery Controller where the scripts are executed
- Run the scripts from a location that has access to the Delivery Controller where the machine catalog was created
- The scripts are supported on the following hypervisor platforms: Citrix XenServer, Microsoft Hyper-V, and VMware ESX

Back up a machine catalog

Use migration-backup.ps1 to back up any machine catalog containing Personal vDisks. The script asks for the name of the machine catalog and connection information for the hypervisor. It then iterates through all of the user-assigned machines in the machine catalog and, for each machine, stores the mapping between the Personal vDisk storage and the assigned user. This information is located in an .xml file. Note that:

- Perform a backup when a change is made to a machine catalog
- You can take a backup while the machines in the machine catalog are active.

The structure of the .xml file is as follows:

```xml
<PVDMigration>
  <hypervisor>
    <type></type>
  </hypervisor>
  <PVD>
    <DiskId></DiskId>
    <DiskName></DiskName>
    <SRName></SRName>
    <SRID></SRID>
    <UserName></UserName>
    <UserSid></UserSid>
    <State></State>
  </PVD>
</PVDMigration>
```

PvDMigration.hypervisor.Type supports VMware ESX, Citrix XenServer, and Microsoft Hyper-V.

PvDMigration.PVD stores information on where the Personal vDisk is stored and the user associated with it.
PvDMigration.PVD.DiskId is the unique identifier of the vDisk on the hypervisor on which the backup was taken.

PvDMigration.PVD.DiskName is the name of the .vhd or .vmdk file.

PvDMigration.PVD.SRName is the name of the storage provider when the backup was taken.

PvDMigration.PVD.SRID is the unique identifier of the storage provider on the hypervisor on which the backup was taken.

PvDMigration.PVD.UserName is the name of the user associated with this vDisk.

PvDMigration.PVD.UserSid is the SID of the user associated with this vDisk.

PvDMigration.PVD.State indicates the state of this vDisk. This can be either backed up or processed. It is backed up after the initial backup. The state changes to processed after the .xml file is used for restoring from the backup.

**Restore a machine catalog**

Important: Before restoring, note the following:

- You can only restore a machine catalog that shares the same master image as that of the backed-up machine catalog.
- You must create a new master image by updating the inventory of the master image that the backed-up machine catalog was created from.

Use migration-restore.ps1 to restore any machine catalog containing Personal vDisks. The script takes the following inputs:

- The .xml file created during the backup process.
- The name of the machine catalog to restore.
- The name of the location where the unattached Personal vDisks are stored. This is listed in the .xml file.
- Hypervisor connection information.

migration-restore.ps1 finds any unassigned machines in the machine catalog and assigns users to them. It also attaches users' Personal vDisks to the machines.

**Scenarios**

Depending on the number of users, you restore their vDisks differently. The rest of this section provides some example scenarios.

**Scenario 1 - Restore a machine catalog and its Personal vDisks using new machine names**

In this scenario, an entire machine catalog and the Personal vDisks attached to the machines in it are restored. The machines are given new names. This scenario may be necessary when your hypervisor or a storage host has failed, or when you migrate users to a new infrastructure.

1. Run migration-backup.ps1 to capture the user-to-Personal-vDisk mapping in the .xml file.
2. Using a backup solution, move or capture the Personal vDisks from the original machine catalog on to a disk:
   - VMware ESX or Microsoft Hyper-V: Personal vDisks are located on the storage specified by the Delivery Controller, in a folder containing the name of the machine to which the vDisk is attached.
   - Citrix XenServer: Personal vDisks are located in the root of the storage specified by the Delivery Controller. The name of each vDisk is a GUID.
3. Restore the Personal vDisks from the original machine catalog using a storage backup solution:
   - ESX or Hyper-V: Locate the vDisks in a new folder of the new storage resource. Alternatively, leave the vDisks in the original path on the new storage resource.
4. Create a Provisioning Services vDisk or a Machine Creation Services snapshot from the master image that you used to create the failed machine catalog.
5. Run Update Inventory from the Start menu on the vDisk or snapshot.
6. Re-create the machine catalog in Studio using a different naming convention as the failed (original) machine catalog. This generates a catalog of new machines, each with a new Personal vDisk, that the XenDesktop database recognizes.
7. Verify that the re-created machine catalog is assigned to the correct Delivery Group.
8. Verify that the Desktop Group is in maintenance mode and the machines in it are shut down.
9. Edit the .xml file generated by the backup script:
   - ESX or Hyper-V: For every PVD section in the file, replace the folder name in DiskName with the location of the restored vDisks. If you restored the vDisks to the original path on the new storage, skip this step.
   - XenServer: Skip this step.
10. On the Delivery Controller, run migration-restore.ps1, specifying the name of the .xml file and the location where the backed-up vDisks are stored.

Scenario 2 - Restore a machine catalog and its Personal vDisks reusing existing machine names

In this scenario, an entire machine catalog and the Personal vDisks attached to the machines in it are restored. Existing (failed) machine names are reused. This scenario may be necessary when your hypervisor or a storage host has failed.

1. Run migration-backup.ps1 to capture the user-to-Personal-vDisk mapping.
2. Using a backup solution, move or capture the Personal vDisks from the original machine catalog on to a disk:
   - ESX or Hyper-V: Personal vDisks are located on the storage specified by the Delivery Controller, in a folder containing the name of the machine to which the vDisk is attached.
   - XenServer: Personal vDisks are located in the root of the storage specified by the Delivery Controller. The name of each vDisk is a GUID.
3. Restore the Personal vDisks from the original machine catalog using a storage backup solution:
   - ESX or Hyper-V: Locate the vDisks in a new folder of the new storage resource.
   - XenServer: Locate the vDisks in the root of the new storage resource.
4. Create a Provisioning Services vDisk or a Machine Creation Services snapshot from the master image that you used to create the failed machine catalog.
5. Run Update Inventory from the Start menu on the vDisk or snapshot.
6. Re-create the machine catalog in Studio using the same naming convention as the failed machine catalog. This generates a catalog of new machines, each with a new Personal vDisk, that the XenDesktop database recognizes.
7. Verify that the re-created machine catalog is assigned to the correct Delivery Group.
8. Verify that the Desktop Group is in maintenance mode and the machines in it are shut down.
9. Edit the .xml file generated by the backup script:
   - ESX or Hyper-V: For every PVD section in the file, replace the folder name in DiskName with the location of the restored vDisks.
   - XenServer: Skip this step.
10. Run the migration-restore.ps1 script on the Delivery Controller with the modified .xml file as an input. The script attaches the vDisks without moving them.
11. Verify the users’ data has been successfully restored.

Scenario 3 - Restore a subset of Personal vDisks in a machine catalog
In this scenario, some, but not all, of the Personal vDisks in a machine catalog have failed and are restored. The virtual machines in the catalog have not failed.

1. Run migration-backup.ps1 to capture the user-to-Personal-vDisk mapping in the .xml file.
2. The .xml file has a PVD section for each user in the machine catalog. For any users whose Personal vDisks do not need restoring, remove the users and their associated sections from the file.
3. Restore the Personal vDisks from the original machine catalog using a backup solution, as described in the one of the other scenarios:
   - To use new machine names, follow Scenario 1.
   - To preserve machine names, follow Scenario 2.
4. Ensure there are enough unassigned machines in the catalog. Add machines if necessary. You need one new machine for each user whose vDisk you want to restore.
5. Verify that the Desktop Group is in maintenance mode and the machines in it are shut down.
6. On the Delivery Controller, run migration-restore.ps1 with the modified .xml file as an input.
7. Verify the users' data has been successfully restored.
Configuration Logging

Jul 26, 2013

Configuration Logging captures Site configuration changes and administrative activities to the Database. You can use the logged content to:

- Diagnose and troubleshoot problems after configuration changes are made; the log provides a breadcrumb trail
- Assist change management and track configurations
- Report administration activity

You set Configuration Logging preferences, display configuration logs, and generate HTML and CSV reports from Citrix Studio. You can filter configuration log displays by date ranges and by full text search results. Mandatory logging, when enabled, prevents configuration changes from being made unless they can be logged. With appropriate permission, you can delete entries from the configuration log. You cannot use the Configuration Logging feature to edit log content.

Configuration Logging uses a PowerShell 2.0 SDK and the Configuration Logging Service. The Configuration Logging Service runs on every Controller in the Site; if one Controller fails, the service on another Controller automatically handles logging requests.

By default, the Configuration Logging feature is enabled, and uses the Database that is created when you create the Site (the Site Configuration Database). Citrix strongly recommends that you change the location of the database used for Configuration Logging as soon as possible after creating a Site. The Configuration Logging Database supports the same high availability features as the Site Configuration Database.

Access to Configuration Logging is controlled through Delegated Administration, with the Edit Logging Preferences and View Configuration Logs permissions.

Configuration logs are localized when they are created. For example, a log created in English will be read in English, regardless of the locale of the reader.

What is logged

Configuration changes and administrative activities initiated from Studio, Director, and PowerShell scripts are logged. Examples of logged configuration changes include working with (creating, editing, deleting assigning):

- Machine Catalogs
- Delivery Groups (including changing power management settings)
- Administrator roles and scopes
- Host resources and connections
- Citrix policies through Studio

Examples of logged administrative changes include:

- Power management of a virtual machine or a user desktop
- Studio or Director sending a message to a user

The following operations are not logged:

- Autonomic operations such as pool management power-on of virtual machines.
- Policy actions implemented through the Group Policy Management Console (GPMC); use Microsoft tools to view logs of those actions.
- Changes made through the registry, direct access of the Database, or from sources other than Studio, Director, or PowerShell.
● When the deployment is initialized, Configuration Logging becomes available when the first Configuration Logging Service instance registers with the Configuration Service. Therefore, the very early stages of configuration are not logged (for example, when the Database schema is obtained and applied, when a hypervisor is initialized).
Manage Configuration Logging

By default, Configuration Logging is enabled, and mandatory logging is disabled.

By default, Configuration Logging uses the database that is created when you create a Site (also known as the Site Configuration Database). Citrix recommends that you change the location of the database used for Configuration Logging (and the database used for the Monitoring Service, which also uses the Site Configuration Database by default) after creating a Site, for the following reasons:

- The backup strategy for the Configuration Logging Database is likely to differ from the backup strategy for the Site Configuration Database.
- The volume of data collected for Configuration Logging (and the Monitoring Service) could adversely affect the space available to the Site Configuration database.
- It splits the single point of failure for the three databases.

For more information, see Change secondary database locations.

The Configuration Logging Database supports the same high availability features as the Site Configuration Database. Review the database guidance in Plan a deployment.

To enable/disable Configuration Logging and mandatory logging

1. From Citrix Studio, select Logging in the left pane.
   Note: Editions that do not support Configuration Logging do not have a Logging node.

2. In the Actions pane, click Preferences. The Configuration Logging dialog box appears, displaying database information and whether Configuration Logging and mandatory logging are enabled or disabled.
   - To enable or disable Configuration Logging:
     - To enable Configuration Logging, select the Enable logging radio button. This is the default setting. If the database cannot be written to, the logging information is discarded, but the operation continues.
     - To disable Configuration Logging, select the Disable logging radio button. If logging was previously enabled, existing logs remain readable with the PowerShell SDK.
   - To enable or disable mandatory logging:
     - To enable mandatory logging, clear the Allow changes when the database is disconnected checkbox. No configuration change or administrative activity that would normally be logged will be allowed unless it can be written in the database used for Configuration Logging.
     - To disable mandatory logging, select the Allow changes when the database is disconnected checkbox. Configuration changes and administrative activities are allowed, even if the database used for Configuration Logging cannot be accessed. This is the default setting.
   The mandatory logging option is available only when Configuration Logging is enabled, that is, when the Enable Configuration Logging radio button is selected. If the Configuration Logging Service fails, and high availability is not in use, mandatory logging is assumed. In such cases, operations that would normally be logged are not performed.

To change the Configuration Logging Database location

1. Create a database server, using a supported SQL Server version.
2. From Studio, select Logging in the left pane.
3. In the Actions pane, click Preferences. The Configuration Logging dialog box appears.
5. Specify the location of the server containing the new database server (using one of the forms in the following table) and the database name.

<table>
<thead>
<tr>
<th>Database type</th>
<th>What to enter</th>
<th>With this database configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standalone or mirror</td>
<td>servername</td>
<td>The default instance is used and SQL Server uses the default port.</td>
</tr>
<tr>
<td></td>
<td>Servername\INSTANCENAME</td>
<td>A named instance is used and SQL Server uses the default port.</td>
</tr>
<tr>
<td></td>
<td>servername,port-number</td>
<td>The default instance is used and SQL Server uses a custom port. (The comma is required.)</td>
</tr>
<tr>
<td>Other</td>
<td>cluster-name</td>
<td>A clustered database.</td>
</tr>
<tr>
<td></td>
<td>availability-group-listener</td>
<td>An Always-On database.</td>
</tr>
</tbody>
</table>

6. If you want Studio to create the database, click OK or Test connection. When prompted, click OK, and Studio will create the database automatically. Studio attempts to access the database using the current Studio user's credentials; if that fails, you are prompted for the database user's credentials. Studio then uploads the database schema to the database. (The credentials are retained only for the database creation time frame.)

7. If you want to create the database manually, click Generate script (or use Get-LogDBSchema in the PowerShell SDK). The generated script includes instructions for manually creating the database. Ensure that the database is empty and that at least one user has permission to access and change the database before uploading the schema.

The Configuration Logging data in the previous database is not imported to the new database. Logs cannot be aggregated from both databases when retrieving logs. The first log entry in the new Configuration Logging Database will indicate that a database change occurred, but it does not identify the previous database.
Display configuration log content

Feb 06, 2013
When initiating configuration changes and administrative activities, Citrix Studio and Citrix Director create high level operations, which are displayed in the upper portion of the center pane in Studio. A high level operation results in one or more service and SDK calls, which are low level operations. When you select a high level operation in the upper portion of the center pane, the lower portion of the center pane displays the low level operations.

If an operation fails before completion, the log operation might not be completed in the Database; for example, a start record will have no corresponding stop record. In such cases, the log indicates that there is missing information. When you display logs based on time ranges, incomplete logs are shown if the data in the logs matches the criteria. For example, if all logs for the last five days are requested and a log exists with a start time in the last five days but has no end time, it is included.

When using a script to call PowerShell cmdlets, if you create a low level operation without specifying a parent high level operation, Configuration Logging will create a surrogate high level operation.

To display configuration log content
From Studio, select Logging in the left pane. By default, the display in the center pane lists the log content chronologically (newest entries first), separated by date.

<table>
<thead>
<tr>
<th>To filter the display by</th>
<th>Complete this action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search results</td>
<td>Enter text in the Search box at the top of the center pane. The filtered display includes the number of search results. To return to the standard logging display, clear the text in the Search box.</td>
</tr>
<tr>
<td>Column heading</td>
<td>Click a column heading to sort the display by that field.</td>
</tr>
<tr>
<td>A date range</td>
<td>Select an interval from the drop down list box, which is next to the Search box at the top of the center pane (today, last 7 days, last 28 days, last three months, last six months).</td>
</tr>
</tbody>
</table>
Generate configuration log reports

Feb 06, 2013

You can generate CSV and HTML reports containing configuration log data.

- The CSV report is a data dump of the logging data from a specified time interval. The hierarchical data in the database is flattened into a single CSV table. No aspect of the data has precedence in the file. No formatting is used and no human readability is assumed. The file (named MyReport) simply contains the data in a universally consumable format. CSV files are often used for archiving data or as a data source for a reporting or data manipulation tool such as Microsoft Excel.
- The HTML report provides a human-readable form of the logging data for a specified time interval. It provides a structured, navigable view for reviewing changes. An HTML report comprises two files, named Summary and Details. Summary lists high level operations: when each operation occurred, by whom, and the outcome. Clicking a Details link next to each operation takes you to the low level operations in the Details file, which provides additional information.

To generate a configuration log report

1. From Citrix Studio, select Logging in the left pane.
2. In the Actions pane, click Create custom report.
3. Select the date range for the report by selecting a predefined interval or specifying a custom time frame.
4. Select the report format: CSV, HTML, or Both CSV and HTML.
5. Browse to the location where the report should be saved.
6. Review your selections on the Summary page and click Finish.
Delete configuration log content

Jun 17, 2013
To delete the configuration log, you must have certain Delegated Administration and SQL Server database permissions.

- **Delegated Administration** — You must have a Delegated Administration role that allows the deployment configuration to be read. The built-in Full administrator role has this permission. A custom role must have Read Only or Manage selected in the Other permissions category.

To create a backup of the configuration logging data before deleting it, the custom role must also have Read Only or Manage selected in the Logging Permissions category.

- **SQL Server database** — You must have a SQL server login with permission to delete records from the database. There are two ways to do this:
  - Use a SQL Server database login with a sysadmin server role, which allows you to perform any activity on the database server. Alternatively, the serveradmin or setupadmin server roles allow you to perform deletion operations.
  - If your deployment requires additional security, use a non-sysadmin database login mapped to a database user who has permission to delete records from the database.
    1. In SQL Server Management Studio, create a SQL Server login with a server role other than 'sysadmin.'
    2. Map the login to a user in the database; SQL Server automatically creates a user in the database with the same name as the login.
    3. In Database role membership, specify at least one of the role members for the database user: ConfigurationLoggingSchema_ROLE or dbowner.
       For more information, see the SQL Server Management Studio documentation.

To delete the configuration logs

1. From Studio, select Logging in the left pane.
2. In the Actions pane, click Delete logs.
3. You are asked if you want to create a backup of the logs before they are deleted. If you choose to create a backup, browse to the location where the backup archive should be saved. The backup will be created as a CSV file.
4. Review your selections on the Summary page, and then click Finish.

After the configuration logs are cleared, the log deletion is the first activity posted to the empty log. That entry provides details about who deleted the logs, and when.
Delegated Administration

Jun 17, 2013

The Delegated Administration model offers the flexibility to match how your organization wants to delegate administration activities, using role and object-based control. Delegated Administration accommodates deployments of all sizes, and allows you to configure more permission granularity as your deployment grows in complexity. Delegated Administration uses three concepts: administrators, roles, and scopes.

- **Administrators** — An administrator represents an individual person or a group of people identified by their Active Directory account. Each administrator is associated with one or more role and scope pairs.

- **Roles** — A role represents a job function, and has defined permissions associated with it. For example, the Delivery Group Administrator role has permissions such as 'Create Delivery Group' and 'Remove Desktop from Delivery Group.' An administrator can have multiple roles for a Site, so a person could be a Delivery Group Administrator and a Machine Catalog Administrator. Roles can be built-in or custom.

The built-in roles are:

<table>
<thead>
<tr>
<th>Role</th>
<th>Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Administrator</td>
<td>Can perform all tasks and operations. A Full Administrator is always combined with the All scope.</td>
</tr>
<tr>
<td>Read Only Administrator</td>
<td>Can see all objects in specified scopes as well as global information, but cannot change anything. For example, a Read Only Administrator with Scope=London can see all global objects (such as Configuration Logging) and any London-scoped objects (for example, London Delivery Groups). However, that administrator cannot see objects in the New York scope (assuming that the London and New York scopes do not overlap).</td>
</tr>
<tr>
<td>Help Desk Administrator</td>
<td>Can view Delivery Groups, and manage the sessions and machines associated with those groups. Can see the Machine Catalog and host information for the Delivery Groups being monitored, and can also perform session management and machine power management operations for the machines in those Delivery Groups.</td>
</tr>
<tr>
<td>Machine Catalog Administrator</td>
<td>Can create and manage Machine Catalogs and provision the machines into them. Can build Machine Catalogs from the virtualization infrastructure, Provisioning Services, and physical machines. This role can manage base images and install software, but cannot assign applications or desktops to users.</td>
</tr>
<tr>
<td>Delivery Group Administrator</td>
<td>Can deliver applications, desktops, and machines; can also manage the associated sessions. Can also manage application and desktop configurations such as policies and power management settings.</td>
</tr>
<tr>
<td>Host Administrator</td>
<td>Can manage host connections and their associated resource settings. Cannot deliver machines, applications, or desktops to users.</td>
</tr>
</tbody>
</table>

In certain XenDesktop editions, you can create custom roles to match the requirements of your organization, and delegate permissions with more detail. You can use custom roles to allocate permissions at the granularity of an action or task in a console.

- **Scopes** — A scope represents a collection of objects. Scopes are used to group objects in a way that is relevant to your
organization (for example, the set of Delivery Groups used by the Sales team). Objects can be in more than one scope; you can think of objects being labeled with one or more scopes. There is one built-in scope: 'All,' which contains all objects. The Full Administrator role is always paired with the All scope.

**Example**

Company XYZ decided to manage applications and desktops based on their department (Accounts, Sales, and Warehouse) and their desktop operating system (Windows 7 or Windows 8). The administrator created five scopes, then labeled each Delivery Group with two scopes: one for the department where they are used and one for the operating system they use.

The following administrators were created:

<table>
<thead>
<tr>
<th>Administrator</th>
<th>Roles</th>
<th>Scopes</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain/fred</td>
<td>Full Administrator</td>
<td>All (the Full Administrator role always has the All scope)</td>
</tr>
<tr>
<td>domain/rob</td>
<td>Read Only Administrator</td>
<td>All</td>
</tr>
<tr>
<td>domain/heidi</td>
<td>Read Only Administrator</td>
<td>All</td>
</tr>
<tr>
<td></td>
<td>Help Desk Administrator</td>
<td>Sales</td>
</tr>
<tr>
<td>domain/warehouseadmin</td>
<td>Help Desk Administrator</td>
<td>Warehouse</td>
</tr>
<tr>
<td>domain/peter</td>
<td>Delivery Group Administrator and Machine Catalog Administrator</td>
<td>Win7</td>
</tr>
</tbody>
</table>

- Fred is a Full Administrator and can view, edit, and delete all objects in the system.
- Rob can view all objects in the Site but cannot edit or delete them.
- Heidi can view all objects and can perform help desk tasks on Delivery Groups in the Sales scope. This allows her to manage the sessions and machines associated with those groups; she cannot make changes to the Delivery Group, such as adding or removing machines.
- Anyone who is a member of the WarehouseAdmins Active Directory security group can view and perform help desk tasks on machines in the Warehouse scope.
- Peter is a Windows 7 specialist and can manage all Windows 7 Machine Catalogs and can deliver Windows 7 applications, desktops, and machines, regardless of which department scope they are in. The administrator considered making Peter a Full Administrator for the Win7 scope; however, she decided against this, because a Full Administrator also has full rights over all objects that are not scoped, such as 'Site' and 'Administrator.'

**How to use Delegated Administration**

Generally, the number of administrators and the granularity of their permissions depends on the size and complexity of the deployment.

- In small or proof-of-concept deployments, one or a few administrators do everything; there is no delegation. In this case, create each administrator with the built-in Full Administrator role, which has the All scope.
- In larger deployments with more machines, applications, and desktops, more delegation is needed. Several administrators
might have more specific functional responsibilities (roles). For example, two are Full Administrators, and others are Help Desk Administrators. Additionally, an administrator might manage only certain groups of objects (scopes), such as machine catalogs. In this case, create new scopes, plus administrators with one of the built-in roles and the appropriate scopes.

- Even larger deployments might require more (or more specific) scopes, plus different administrators with unconventional roles. In this case, edit or create additional scopes, create custom roles, and create each administrator with a built-in or custom role, plus existing and new scopes.

For flexibility and ease of configuration, you can create new scopes when you create an administrator. You can also specify scopes when creating or editing Machine Catalogs or Hosts.
Administrators

Apr 15, 2013

When you create a Site as a local administrator, your user account automatically becomes a Full Administrator with full permissions over all objects. After a Site is created, local administrators have no special privileges.

By default, an administrator is enabled. You can disable an administrator, if needed. This might be necessary if you are configuring a new administrator now, but that person will not be beginning administration duties until some time in the future. For existing enabled administrators, you might want to disable several of them while you are reorganizing your object/scopes, then re-enable them when you are ready to go live with the updated configuration.

Important: To prevent the system from becoming unusable or unmanageable, there must always be at least one Full Administrator; therefore, you cannot disable or delete that administrator.

You can produce a printable HTML report listing all the roles, scopes, and permissions an administrator has.

To create an administrator

The Full Administrator role always has the All scope; you cannot change this.

1. In Studio, click Configuration > Administrators in the left pane, then click the Administrators tab in the middle pane. A list of existing administrators appears.
2. In the Actions pane, click Create new Administrator.
3. Type the name of the administrator user account or browse to it.
4. To specify the objects the administrator can access, select a scope. You can select an existing scope or create a new one (see To create a scope).
5. To define the permissions the administrator has for the scoped objects, select a role.
6. A summary of the new administrator's details appears. By default, a new administrator is enabled. To disable the new administrator, clear the Enable Administrator checkbox.
   To create a report listing all the permissions the administrator will have, click Create a full permissions report for this Administrator (HTML format).
7. To create the administrator, click Finish.

To copy an administrator

You can create an administrator by copying the details of an existing administrator. Unless you change them, the new administrator will have the same role/scope pairs as the copied administrator.

1. In Studio, click Configuration > Administrators in the left pane, then click Administrators tab in the middle pane. A list of existing administrators appears.
2. In the middle pane, select the administrator you want to copy, then click Copy Administrator in the Actions pane. The role/scope pairs for the administrator you are copying appear.
3. Type the name of the new administrator's user account or browse to it.
4. You can edit or delete any of the role/scope pairs, and you can add new ones. By default the new administrator will be enabled. To disable the new administrator, clear Enable Administrator.
   To create a report listing all the permissions the new administrator will have, click View full permissions report.
5. To create the new administrator, click Save.
To edit an administrator

1. In Studio, click Configuration > Administrators in the left pane, then click the Administrators tab in the middle pane. A list of existing administrators appears.

2. In the middle pane, select the administrator you want to edit, then click Edit Administrator in the Actions pane. The details of that administrator appear. You can update them as follows:
   - To add a role/scope pair, click Add. You are led through the steps for adding scopes and roles in the same way as when you create a new administrator.
   - To edit a role/scope pair, click the row containing the pair you want to edit, then click Edit. A list of scopes appears; the scope currently paired with the role is selected. You can either change the objects for which the administrator has permissions by selecting a different scope, or you can change the permissions that the administrator has for the objects by selecting a different role:
     - To select a different scope, click it, then click Save.
     - To create a new scope, click Create new Scope (see To create a scope).
     - To select a different role, click Role in the left pane. A list of roles appears; the role currently paired with the scope is selected. Click the new role, then click Save.
     - To create a new role, click Create new Role (see To create a role).
   - To delete a role/scope pair, click the row containing the pair you want to delete, then click Delete. This deletes only the relationship between the role and the scope for this administrator; it does not delete either the role or the scope, nor does it affect any other administrator who might also be configured with this role/scope pair.
     - To enable or disable the administrator, select or clear Enable Administrator. You cannot disable the administrator if it would result in there being no enabled Full Administrator.

To delete an administrator

You cannot delete an administrator if it would result in there being no Full Administrator.

1. In Studio, click Configuration > Administrators in the left pane. A list of existing administrators appears.

2. In the middle pane, select the administrator you want to delete, then click Delete Administrator. The scopes and roles associated with this administrator are displayed, so that you can check the impact of the deletion before confirming it.

To create a resultant set of permissions (RSOP) report

You can produce and save a printable HTML report listing all the permissions an administrator has. The report shows the role/scope pairs associated with the administrator, then lists the individual permissions for each type of object (for example delivery groups and machine catalogs).

1. In Studio, click Configuration > Administrators in the left pane. A list of existing administrators appears.

2. In the middle pane, select the administrator, then click Create RSOP Report.

You can also create a report when creating, copying, or editing an administrator.
Roles

Jan 24, 2013
You can view details about all roles. You can copy a built-in role. You can create, copy, edit, or delete a custom role.

Role names can contain up to 64 Unicode characters; they cannot contain the following characters: \ (backslash), / (forward slash), : (semicolon), # (pound sign), , (comma), * (asterisk), ? (question mark), = (equal sign), < (left arrow), > (right arrow), | (pipe), [ ] (left or right bracket), ( ) (left or right parenthesis), " (quotation marks), and ' (apostrophe).

Descriptions can contain up to 256 Unicode characters.

Note: Only certain editions support custom roles. Editions that do not support custom roles do not have related entries in the Actions pane.

To view role details

1. In Studio, click Configuration > Administrators in the left pane, then click the Roles tab in the middle pane. A list of existing roles appears.
2. Select the role you want to display. The lower portion of the middle pane lists the object types and associated permissions for the role. Click the Administrators tab in the lower pane to display a list of administrators who currently have this role.

To create a custom role

1. In Studio, click Configuration > Administrators in the left pane, then click the Roles tab in the middle pane. A list of existing roles appears.
2. In the Actions pane, click Create new Role.
3. Type a name and description for the role.
4. For each object type that you want this role to have permissions for, select the object type, then select the permissions.
5. Click Save.

To copy a role

You can create a role by copying the details of an existing role. You can copy a role only if you are allowed to create a custom role.

1. In Studio, click Configuration > Administrators in the left pane, then click the Roles tab in the middle pane. A list of existing roles appears.
2. In the middle pane, select the role you want to copy, then click Copy Role in the Actions pane.
3. Type a name and a description for the new role.
4. Edit the copied role's object types and permissions, as necessary.
5. Click Save.

To edit a custom role

Before you edit a custom role, ensure that the changes will not adversely impact administrators with this role. You cannot edit a built-in role.

1. In Studio, click Configuration > Administrators in the left pane, then click the Roles tab in the middle pane. A list of existing roles appears.
2. In the middle pane, select the role you want to edit, then click Edit Role in the Actions pane.
3. Update the name, description, object types, and permissions, as necessary.
4. Click Save.

To delete a custom role

You cannot delete a custom role if any administrator is using it. You cannot delete a built-in role.

1. In Studio, click Configuration > Administrators in the left pane, then click the Roles tab in the middle pane. A list of existing roles appears.
2. In the middle pane, select the role you want to delete, then click Delete Role in the Actions pane.
3. When prompted, confirm the deletion.
Scopes

Jun 17, 2013

When you create a Site, the only available scope is the 'All' scope, which cannot be deleted.

You can create scopes using the procedure in this topic. You can also create scopes when you create an administrator; each administrator must be associated with at least one role and scope pair. When you are creating or editing desktops, machine catalogs, applications, or hosts, you can add them to an existing scope; if you do not add them to a scope, they remain part of the 'All' scope.

Site creation cannot be scoped, nor can Delegated Administration objects (scopes and roles). However, objects you cannot scope are included in the 'All' scope. (Full Administrators always have the All scope.) Machines, power actions, desktops, and sessions are not directly scoped; administrators can be allocated permissions over these objects through the associated machine catalogs or Delivery Groups.

Scope names can contain up to 64 Unicode characters; they cannot include the following characters: \ (backslash), / (forward slash), : (semicolon), # (pound sign), , (comma), * (asterisk), ? (question mark), = (equal sign), < (left arrow), > (right arrow), | (pipe), [ ] (left or right bracket), ( ) (left or right parenthesis), " (quotation marks), and ' (apostrophe). Descriptions can contain up to 256 Unicode characters.

To create a scope

1. In Studio, click Configuration > Administrators in the left pane, then click the Scopes tab in the middle pane. A list of existing scopes appears.
2. In the Actions pane, click Create new Scope.
3. Type a name (for example, 'Sales') and a description (for example, 'Delivery Groups used by Sales') for the scope.
4. Select object types or specific objects:
   1. To select all objects of a particular type (for example, Delivery Groups), select the object type.
   2. To select specific objects, expand the type, then select the individual objects (for example, individual Delivery Groups used by the Sales team).
5. Click Save.

To copy a scope

You can create a scope by copying an existing scope.

1. In Studio, click Configuration > Administrators in the left pane, then click the Scopes tab in the middle pane. A list of existing scopes appears.
2. In the middle pane, select the scope you want to copy, then click Copy Scope in the Actions pane.
3. Type a name and a description for the new scope.
4. Edit the copied scope's object types and objects, as necessary.
5. Click Save.

To edit a scope

1. In Studio, click Configuration > Administrators in the left pane. then click the Scopes tab in the middle pane. A list of existing scopes appears.
2. In the middle pane, select the scope you want to edit, then click Edit Scope in the Actions pane.
3. Update the name, description, object types, and objects, as necessary.
Important: Removing objects from a scope can make those objects inaccessible to the administrator who removed them. If the edited scope is paired with one or more roles, ensure that the updates you make to the scope do not make any role/scope pair unusable.

4. Click Save.

To delete a scope

You cannot delete a scope if any administrator is using it.

1. In Studio, select Configuration > Administrators in the left pane, then click the Scopes tab in the middle pane. A list of existing scopes appears.
2. In the middle pane, select the scope you want to delete, then click Delete Scope in the Actions pane.
3. When prompted, confirm the deletion.
Reports

Feb 14, 2013

You can produce the following Delegated Administration reports:

- An HTML report listing the permissions for an individual administrator. This is called a resultant set of permissions (RSOP) report, and is requested through Citrix Studio.
- An HTML or CSV report that maps all built-in and custom roles to permissions. This report is created by running a PowerShell script.

To create an RSOP report

This HTML report shows the role/scope pairs associated with an administrator, and lists the individual permissions for each type of object (for example, Delivery Groups and Machine Catalogs).

1. In Studio, click Configuration > Administrators in the left pane. A list of existing administrators appears.
2. In the middle pane, select an administrator, then click Create RSOP Report.

You can also request this report when creating, copying, or editing an administrator.

To create a role-to-permission mapping report

Use the OutputPermissionMapping.ps1 PowerShell script to build an HTML or CSV table that lists the permissions associated with each built-in and custom role.

To run this script, you must be a Full Administrator, a Read Only Administrator, or a custom administrator with permission to read roles. The script is located in: Program Files\Citrix\DelegatedAdmin\SnapIn\Citrix.DelegatedAdmin.Admin.V1\Scripts\.

Syntax:

OutputPermissionMapping.ps1 [-Help] [-Csv] [-Path <string>] [-AdminAddress <string>] [-Show] [<CommonParameters>]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Help</td>
<td>Displays script help.</td>
</tr>
<tr>
<td>-Csv</td>
<td>Specifies CSV output. Default = HTML</td>
</tr>
<tr>
<td>-Path &lt;string&gt;</td>
<td>Specifies where to write the output. Default = stdout</td>
</tr>
<tr>
<td>-AdminAddress &lt;string&gt;</td>
<td>Specifies the IP address or host name of the Delivery Controller to connect to. Default = localhost</td>
</tr>
<tr>
<td>-Show</td>
<td>(Valid only when the -Path parameter is also specified) When you write the output to a file, this parameter causes the output to be opened in an appropriate program, such as a web browser.</td>
</tr>
<tr>
<td>&lt;CommonParameters&gt;</td>
<td>Common parameters supported by this script: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, and OutVariable. For details, see the Microsoft documentation.</td>
</tr>
</tbody>
</table>

The following example writes an HTML table to a file named Roles.html and opens the table in a web browser.
& "$env:ProgramFiles\Citrix\DelegatedAdmin\SnapIn\Citrix.DelegatedAdmin.Admin.V1\Scripts\OutputPermissionMapping.ps1" -Path Roles.html –Show
The following example writes a CSV table to a file named Roles.csv. The table is not displayed. & "$env:ProgramFiles\Citrix\DelegatedAdmin\SnapIn\Citrix.DelegatedAdmin.Admin.V1\Scripts\OutputPermissionMapping.ps1" –CSV -Path Roles.csv
From a Windows command prompt, the preceding example command is:
powershell -command "& '%ProgramFiles%\Citrix\DelegatedAdmin\SnapIn\Citrix.DelegatedAdmin.Admin.V1\Scripts\OutputPermissionMapping.ps1' -CSV -Path Roles.csv"
IPv6

Mar 12, 2015
This release supports pure IPv4, pure IPv6, and dual-stack deployments that use overlapping IPv4 and IPv6 networks.

IPv6 communications are controlled with two Virtual Delivery Agent (VDA) connection-related Citrix policy settings:
- A primary setting that enforces the use of IPv6: Only use IPv6 Controller registration.
- A dependent setting that defines an IPv6 netmask: Controller registration IPv6 netmask.

When the Only use IPv6 Controller registration policy setting is enabled, VDAs register with a Delivery Controller for incoming connections using an IPv6 address.

Dual-stack IPv4/IPv6 deployment

The following figure illustrates a dual-stack IPv4/IPv6 deployment. In this scenario, a worker is a VDA installed on a hypervisor or on a physical system, and is used primarily to enable connections for applications and desktops. Components that support dual IPv6 and IPv4 are running on operating systems that use tunneling or dual protocol software.

These Citrix products, components, and features support only IPv4:
- Publishing App-V
- Provisioning Services
- XenServer Version 6.x
- VDAs not controlled by the Only use IPv6 Controller registration policy setting
- Earlier versions of XenApp, XenDesktop, and EdgeSight

In this deployment:
- If a team frequently uses an IPv6 network and the administrator wants them to use IPv6 traffic, the administrator will publish IPv6 desktops and applications for those users based on a worker image or Organizational Unit (OU) that has the primary IPv6 policy setting turned on (that is, Only use IPv6 Controller registration is enabled).
If a team frequently uses an IPv4 network, the administrator will publish IPv4 desktops and applications for those users based on a worker image or OU that has the primary IPv6 policy setting turned off (that is, Only use IPv6 Controller registration is disabled), which is the default.

**Deployment considerations**

If your environment contains both IPv4 and IPv6 networks, you will need separate Delivery Group configurations for the IPv4-only clients and for the clients who can access the IPv6 network. Consider using naming, manual Active Directory group assignment, or Smart Access filters to differentiate users.

Reconnection to a session may fail if the connection is initiated on an IPv6 network, and then attempts are made to connect again from an internal client that has only IPv4 access.

**Pure IPv6 deployment**

The following figure illustrates a pure IPv6 deployment. In this scenario:

- The components are running on operating systems configured to support an IPv6 network.
- The primary Citrix policy setting (Only use IPv6 Controller registration) is enabled for all VDAs; they must register with the Controller using an IPv6 address.
Policy settings for IPv6

Jun 17, 2013

Two Citrix policy settings affect support for a pure IPv6 or dual stack IPv4/IPv6 implementation. Configure the following connection-related policy settings:

- **Only use IPv6 Controller registration** — Controls which form of address the Virtual Delivery Agent (VDA) uses to register with the Delivery Controller. Default = Disabled
  
  - When the VDA communicates with the Controller, it uses a single IPv6 address chosen in the following precedence: global IP address, Unique Local Address (ULA), link-local address (only if no other IPv6 addresses are available).
  
  - When disabled, the VDA registers and communicates with the Controller using the machine's IPv4 address.

- **Controller registration IPv6 netmask** — A machine can have multiple IPv6 addresses; this policy setting allows administrators to restrict the VDA to only a preferred subnet (rather than a global IP, if one is registered). This setting specifies the network where the VDA will register: the VDA registers only on the first address that matches the specified netmask. This setting is valid only if the Only use IPv6 Controller registration policy setting is enabled. Default = Empty string

Important: Use of IPv4 or IPv6 by a VDA is determined solely by these policy settings. In other words, to use IPv6 addressing, the VDA must be controlled by a Citrix policy with the Only use IPv6 Controller registration setting enabled.
Configure time zone settings

Oct 11, 2012

By default, when non-privileged users connect to Windows XP desktops, they see the time zone of the system running the desktop instead of the time zone of their own user device. This does not apply to Windows Vista or Windows 7, which have a separate time zone privilege.

To allow Windows XP users to see their local time you need to give them rights to:

- Change the time on the system on which the desktop is running. To do this, set up a Group Policy with rights given to non-privileged users to change system time settings. For further information about how to do this, see http://msdn2.microsoft.com/en-us/library/ms813808.aspx.
- Change the time zone registry area. For information about how to do this, see http://support.microsoft.com/kb/300022/.

After you do this, users who connect to Windows XP desktops see their local time zone reflected in the desktop. When they log off or disconnect, the time zone of the desktop is reset to what it was before they logged on.

You can configure time zone settings through Citrix policies. Use the Use local time of client policy setting in the Time Zone Control section of the ICA Policy Settings folder.
Configure connection timers

Oct 11, 2012

You can configure three connection timers:

- A maximum connection timer. This setting determines the maximum duration of an uninterrupted connection between a user device and a virtual desktop. Use the Session connection timer and Session connection timer interval policy settings to configure this.

- A connection idle timer. This setting determines how long an uninterrupted user device connection to a virtual desktop will be maintained if there is no input from the user. Use the Session idle timer and Session idle timer interval policy settings to configure this.

- A disconnect timer. This setting determines how long a disconnected, locked virtual desktop can remain locked before the session is logged off. Use the Disconnected session timer and Disconnected session timer interval policy settings to configure this.

If you need to update any of these settings, ensure that settings are consistent across your deployment.
Configure USB support

Dec 03, 2014

HDX USB device redirection enables redirection of USB devices to and from a user device. For example, a user can connect a flash drive to a local computer and access it remotely from within a virtual desktop or a desktop hosted application. During a session, users can plug and play devices, including Picture Transfer Protocol (PTP) devices such as digital cameras, Media Transfer Protocol (MTP) devices such as digital audio players or portable media players, and point-of-sale (POS) devices.

Double-hop USB is not supported for desktop hosted application sessions.

USB redirection is available for the following Receivers:
- Receiver for Windows
- Receiver for Linux

By default, USB redirection is allowed for certain classes of USB devices, and denied for others. See the Receiver documentation for a list. You can restrict the types of USB devices made available to a virtual desktop by updating the list of USB devices supported for redirection, as described later in this topic.

Important: In environments where security separation between the user device and server is needed, provide guidance to users about the types of USB devices to avoid.

Optimized virtual channels are available to redirect most popular USB devices, and provide superior performance and bandwidth efficiency over a WAN. The level of support provided depends on the Receiver installed on the user device; see the Receiver documentation for support information. Optimized virtual channels are usually the best option, especially in high latency environments.

Note: For USB redirection purposes, XenDesktop handles a SMART board the same as a mouse.

XenDesktop does not support Generic USB Redirection of devices connected to USB 3.0 ports. XenDesktop supports use of optimized virtual channels with USB 3.0 devices and USB 3.0 ports, such as a CDM virtual channel used to view files on a camera or to provide audio to a headset. XenDesktop also supports Generic USB Redirection of USB 3.0 devices connected to a USB 2.0 port.

In XenDesktop, specialty devices for which there is no optimized virtual channel are supported by falling back to a Generic USB virtual channel that provides raw USB redirection. For information on USB devices tested with XenDesktop, see http://support.citrix.com/article/ctx123569.

Some advanced device-specific features, such as Human Interface Device (HID) buttons on a webcam, may not work as expected with the optimized virtual channel; if this is an issue, use the Generic USB virtual channel.

Certain devices are not redirected by default, and are only available to the local session. For example, it would not be appropriate to redirect a network interface card that is attached to the user device's system board by internal USB.

The following Citrix policy settings control USB support:

- Client USB device redirection — The default is Prohibited.
- Client USB device redirection rules — Rules only apply to devices using Generic USB redirection. Therefore, rules do not apply to devices using specialized or optimized redirection, such as CDM.
- Client USB Plug and Play device redirection — The default is Allowed, to permit plug-and-play of PTP, MTP, and POS devices in a user session.
- Client USB device redirection bandwidth limit — The default is 0 (zero, which means no maximum).
- Client USB device redirection bandwidth limit percent — The default is 0 (zero, which means no maximum).
1. Add the Client USB device redirection setting to a policy and set its value to Allowed.
2. (Optional) To update the list of USB devices available for remoting, add the Client USB device redirection rules setting to a policy and specify the USB policy rules, as explained later in this topic.
3. Enable USB support when you install Receiver on user devices. For configuration information, refer to the Receiver documentation. If you specified USB policy rules for the Virtual Delivery Agent in the previous step, specify those same policy rules for Receiver.
   Note: For thin clients, consult the manufacturer for details of USB support and any required configuration.

USB devices are automatically redirected when USB support is enabled and the USB user preference settings are set to automatically connect USB devices. USB devices are also automatically redirected when operating in Desktop Appliance mode and the connection bar is not present. In some instances, however, you might not want to automatically redirect all USB devices. For more information, see CTX123015.

Users can explicitly redirect devices that are not automatically redirected by selecting them from the USB device list. To prevent USB devices from ever being listed or redirected, you can specify device rules on the client and the VDA, as explained below.

You can update the range of USB devices available for remoting by specifying USB device redirection rules for both Receiver and the VDA to override the default USB policy rules.

- Edit the user device registry. An Administrative template (ADM file) is included on the installation media so you can change the user device through Active Directory Group Policy: dvd root \os\lang\Support\Configuration\icaclient_usb.adm.
- Edit the administrator override rules in the VDA registry on the Server OS machines. An ADM file is included on the installation media so you can change the VDA through Active Directory Group Policy: dvd root \os\lang\Support\Configuration\vda_usb.adm.

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

The product default rules are stored in HKLM\SOFTWARE\Citrix\PortICA\GenericUSB\DeviceRules. Do not edit these product default rules. Instead, use them as a guide for creating administrator override rules as explained below. The GPO overrides are evaluated before the product default rules.

The administrator override rules are stored in HKLM\SOFTWARE\Policies\Citrix\PortICA\GenericUSB\DeviceRules. GPO policy rules take the format {Allow:|Deny:} followed by a set of tag=value expressions separated by white space. The following tags are supported:

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VID</td>
<td>Vendor ID from the device descriptor</td>
</tr>
<tr>
<td>PID</td>
<td>Product ID from the device descriptor</td>
</tr>
<tr>
<td>REL</td>
<td>Release ID from the device descriptor</td>
</tr>
<tr>
<td>Class</td>
<td>Class from either the device descriptor or an interface descriptor; see the USB Web site at</td>
</tr>
</tbody>
</table>
### Tag Description

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SubClass</td>
<td>Subclass from either the device descriptor or an interface descriptor</td>
</tr>
<tr>
<td>Prot</td>
<td>Protocol from either the device descriptor or an interface descriptor</td>
</tr>
</tbody>
</table>

When creating new policy rules, note the following:

- Rules are case-insensitive.
- Rules may have an optional comment at the end, introduced by #. A delimiter is not required, and the comment is ignored for matching purposes.
- Blank and pure comment lines are ignored.
- White space is used as a separator, but cannot appear in the middle of a number or identifier. For example, `Deny: Class = 08 SubClass=05` is a valid rule, but `Deny: Class=0 Sub Class=05` is not.
- Tags must use the matching operator =. For example, `VID=1230`.
- Each rule must start on a new line or form part of a semicolon-separated list.

**Important:** If you are using the ADM template file, you must create rules on a single line, as a semicolon-separated list.

When working with optimized devices such as mass storage, you usually redirect the device using the specialized CDM channel rather than with policy rules. However, if either of the following conditions exist, the optimized device is available in the device list in the desktop viewer for Generic USB redirection:

- Auto redirection for storage device is set (for example, `AutoRedirectStorage = 1`); for more information, see [CTX123015](http://support.citrix.com/article/CTX123015).
- Simplify device connections for me is not selected; for more information, see [CTX136716](http://support.citrix.com/article/CTX136716).

**Examples:**

- The following example shows an administrator-defined USB policy rule for vendor and product identifiers:

  ```
  Allow: VID=046D PID=C626 # Allow Logitech SpaceNavigator 3D Mouse
  Deny: VID=046D # Deny all Logitech products
  ```

- The following example shows an administrator-defined USB policy rule for a defined class, sub-class, and protocol:

  ```
  Deny: Class=EF SubClass=01 Prot=01 # Deny MS Active Sync devices
  Allow: Class=EF SubClass=01 # Allow Sync devices
  Allow: Class=EF # Allow all USB-Miscellaneous devices
  ```

By default, USB devices are automatically redirected when USB support is enabled and the USB user preference settings are set to automatically connect USB devices. USB devices are also automatically redirected for Desktop Appliance sites or desktop hosted applications. In some instances, however, you may not want to automatically redirect all USB devices. For more information, see [http://support.citrix.com/article/CTX123015](http://support.citrix.com/article/CTX123015).

Desktop Viewer users can redirect devices that are not automatically redirected by selecting them from the USB device list. To prevent USB devices from being listed or redirected, you can specify device rules on the user device and the VDA, as explained below.

You can update the range of USB devices available for remoting by specifying USB device redirection rules for both Receiver and the VDA to override the default USB policy rules. Device rules are enforced for both Receiver and the VDA. Be sure to change both so that device remoting works as you intend.

- Edit the user device registry (or the `.ini` files in the case of the Receiver for Linux). For information about how to do this, see the Receiver documentation in eDocs. An Administrative template (ADM file) is included on the installation media so
you can change the user device through Active Directory Group Policy: dvd root \os\lang\Support\Configuration\icaclient_usb.adm.

- Edit the administrator override rules in the VDA registry on the Server OS Machine(s). Information about how to do this is included in the rest of this section. An ADM file is included on the installation media so you can change the VDA through Active Directory Group Policy: dvd root \os\lang\Support\Configuration\vda_usb.adm.

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

The product default rules are stored in HKLM\SOFTWARE\Citrix\PortICA\GenericUSB\DeviceRules.

Note: Do not edit these product default rules. Instead, use them as a guide for creating administrator override rules as explained below. The GPO overrides are evaluated before the product default rules. The administrator override rules are stored in HKLM\SOFTWARE\Policies\Citrix\PortICA\GenericUSB\DeviceRules.

GPO policy rules take the format {Allow:|Deny:} followed by a set of tag=value expressions separated by white space. The following tags are supported:

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VID</td>
<td>Vendor ID from the device descriptor</td>
</tr>
<tr>
<td>PID</td>
<td>Product ID from the device descriptor</td>
</tr>
<tr>
<td>REL</td>
<td>Release ID from the device descriptor</td>
</tr>
<tr>
<td>Class</td>
<td>Class from either the device descriptor or an interface descriptor; see the USB Web site at <a href="http://www.usb.org/">http://www.usb.org/</a> for available USB Class Codes</td>
</tr>
<tr>
<td>SubClass</td>
<td>Subclass from either the device descriptor or an interface descriptor</td>
</tr>
<tr>
<td>Prot</td>
<td>Protocol from either the device descriptor or an interface descriptor</td>
</tr>
</tbody>
</table>

When creating new policy rules, note the following:
- Rules are case-insensitive.
- Rules may have an optional comment at the end, introduced by #. A delimiter is not required, and the comment is ignored for matching purposes.
- Blank and pure comment lines are ignored.
- White space is used as a separator, but cannot appear in the middle of a number or identifier. For example, Deny: Class = 08 SubClass=05 is a valid rule, but Deny: Class=0 SubClass=05 is not.
- Tags must use the matching operator =. For example, VID=1230.
- Each rule must start on a new line or form part of a semicolon-separated list.

Important: If you are using the ADM template file, you must create rules on a single line, as a semicolon-separated list.

Use rules with optimized devices

When working with optimized devices, such as mass storage, you usually redirect the device using the specialized CDM channel, rather than with policy rules. However, if either of the following conditions exist, the optimized device is available in the device list in the desktop viewer for Generic USB redirection:

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.416
Auto redirection for storage device is set (for example, AutoRedirectStorage = 1)
For more information about this setting, see How to Configure Automatic Redirection of USB Devices.

Simplify device connections for me is not selected.
For more information about this setting, see USB Device do not Appear in Virtual Desktop.

Examples

This example shows an administrator-defined USB policy rule for vendor and product identifiers:

Allow: VID=046D PID=C626 # Allow Logitech SpaceNavigator 3D Mouse
Deny: VID=046D # Deny all Logitech products

This example shows an administrator-defined USB policy rule for a defined class, sub-class, and protocol:

Deny: Class=EF SubClass=01 Prot=01 # Deny MS Active Sync devices
Allow: Class=EF SubClass=01 # Allow Sync devices
Allow: Class=EF # Allow all USB-Miscellaneous devices

Users can connect a USB device before or after starting a virtual session.

When using Receiver for Windows, the following apply:

- Devices connected after a session starts immediately appear in the USB menu of the Desktop Viewer.
- If a USB device is not redirecting properly, sometimes you can resolve the problem by waiting to connect the device until after the virtual session has started.
- To avoid data loss, use the Windows Safe removal menu before removing the USB device.
Support for USB mass storage devices

Nov 12, 2014

For mass storage devices only, remote access is also available through client drive mapping, where the drives on the user device are automatically mapped to drive letters on the virtual desktop when users log on. The drives are displayed as shared folders with mapped drive letters. To configure client drive mapping, use the Client removable drives setting in the File Redirection Policy Settings section of the ICA Policy Settings.

The main differences between the two types of remoting policy are:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Client drive mapping</th>
<th>Generic USB redirection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enabled by default</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Read-only access configurable</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Safe to remove device during a session</td>
<td>No</td>
<td>Yes, provided users follow operating system recommendations for safe removal</td>
</tr>
</tbody>
</table>

If both Generic USB and the client drive mapping policies are enabled and a mass storage device is inserted either before or after a session starts, it will be redirected using client drive mapping. When both Generic USB and the client drive mapping policies are enabled and a device is configured for automatic redirection (see http://support.citrix.com/article/CTX123015) and a mass storage device is inserted either before or after a session starts, it will be redirected using Generic USB.
Client folder redirection

Sep 27, 2013

Client folder redirection changes the way client-side files are accessible on the host-side session. When you enable only client drive mapping on the server, client-side full volumes are automatically mapped to the sessions as Universal Naming Convention (UNC) links. When you enable client folder redirection on the server and the user configures it on the user device, the portion of the local volume specified by the user is redirected.

Only the user-specified folders appear as UNC links inside sessions instead of the complete file system on the user device. If you disable UNC links through the registry, client folders appear as mapped drives inside the session.

Important: Client folder redirection is supported on Windows Desktop Machines only.

Server platforms supported for client folder redirection:

- Windows Server 2008 R2 SP1 (64-bit)
- Windows Server 2012 (64-bit)
- Windows Server 2012 R2 (64-bit)

Client platforms supported for client folder redirection with the latest Citrix Receiver:

- Windows XP (32-bit and 64-bit)
- Windows 7 (32-bit and 64-bit)
- Windows 8 (32-bit and 64-bit)
- Windows 8.1 (32-bit and 64-bit)
Enable and configure client folder redirection

Oct 21, 2013
For client folder redirection to be enabled and configured, you enable it on the server and the user, on the user device, specifies the folders to be redirected. The user application to specify the client folder options is installed as part of the version of the Citrix Receiver supplied with this release.

Important: Client folder redirection is supported on Windows Desktop Machines only.

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

1. Create a key: HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Citrix\Client Folder Redirection
2. Create a REG_DWORD value.
   - Name: CFROnlyModeAvailable
   - Type: REG_DWORD
   - Data: Set it to 1

Make sure you have the most recent version of Citrix Receiver installed.

1. Start CtxCFRUI.exe from the Receiver installation directory.
2. Select the Custom radio button and add, edit, or remove folders.
3. Disconnect and reconnect your sessions for the setting to take effect.
Monitor

Director is a Web-based tool that enables IT support and help desk teams to monitor a XenDesktop environment, troubleshoot issues before they become system-critical, and perform support tasks for end users.

CtxPvdDiag.exe is a diagnostic tool that enables you to monitor changes made to Personal vDisks.
Monitor environments with Director

Mar 21, 2015

Director provides different views of the interface tailored to particular administrators. XenDesktop permissions determine what is displayed and the commands available.

For example, help desk administrators see an interface tailored to help desk tasks. Director allows help desk administrators to search for the user reporting an issue and display activity associated with that user, such as the status of the user's applications and processes. They can resolve issues quickly by performing actions such as ending an unresponsive application or process, shadowing operations on the user's machine, restarting the machine, or resetting the user profile.

In contrast, full administrators see and manage the entire site and can perform commands for multiple users and machines. The Dashboard provides an overview of the key aspects of a deployment, such as the status of sessions, user logons, and the site infrastructure. Information is updated every minute. If issues occur, details appear automatically about the number and type of failures that have occurred.

Director is installed by default as a website on the Delivery Controller. For prerequisites and other details, see the System requirements topic for this release.

This release of Director is not compatible with XenApp deployments or XenDesktop deployments earlier than 7. The Director monitoring console can monitor and manage only XenDesktop 7.x Sites.

When Director is used in an environment containing more than one XenDesktop 7.x Site, be sure to synchronize the system clocks on all the servers where Controllers, Director, and other core XenDesktop components are installed. Otherwise, the Sites might not display correctly in Director.

Tip: If you intend to monitor XenApp 6.5 and XenDesktop 7.x Sites, Citrix recommends installing Director 7 on a separate server from the Director console that is used to monitor XenApp 6.5 sites. For more information, see Important information for XenApp administrators.

Important: To protect the security of user names and passwords sent using plain text through the network, Citrix strongly recommends that you allow Director connections using only HTTPS, and not HTTP. Certain tools are able to read plain text user names and passwords in HTTP (unencrypted) network packets, which creates a security risk for users.

To log on to Director, administrators with permissions for Director must be Active Directory domain users and must have the following rights:

- Read rights in all Active Directory forests to be searched (see Advanced configuration).
- Configured Delegated Administrator roles (see Delegated Administration and Director).
- To shadow users, administrators must be configured using a Microsoft group policy for Windows Remote Assistance. In addition:
  - When installing VDAs, ensure the Windows Remote Assistance feature is enabled on all user devices (selected by default).
  - When you install Director on a server, ensure that Windows Remote Assistance is installed (selected by default). However, it is disabled on the server by default. The feature does not need to be enabled for Director to provide assistance to end users. Citrix recommends leaving the feature disabled to improve security on the server.
  - To enable administrators to initiate Windows Remote Assistance, grant them the required permissions by using the
appropriate Microsoft Group Policy settings for Remote Assistance. For information, see CTX127388: How to Enable Remote Assistance for Desktop Director.

- For user devices with VDAs earlier than 7, additional configuration is required. See Configure permissions for VDAs earlier than 7.

Install Director using the installer, which checks for prerequisites, installs any missing components, sets up the Director website, and performs basic configuration. The default configuration provided by the installer handles typical deployments. If Director was not included during installation, use the installer to add Director. To add any additional components, rerun the installer and select the components to install. For information on using the installer, see the Installation topics. Citrix recommends that you install using the product installer only, not the .MSI file.

When Director is installed on the Controller, it is automatically configured with localhost as the server address, and Director communicates with the local controller by default.

To install Director on a dedicated server that is remote from a Controller, you are prompted to enter the FQDN or IP address of a Controller. Director communicates with that specified Controller by default. Specify only one Controller address for each Site that you will monitor. Director automatically discovers all other Controllers in the same Site and falls back to those other Controllers if the Controller you specified fails.

Note: Director does not load balance between Controllers.
To secure the communications between the browser and the Web server, Citrix recommends that you implement SSL on the IIS website hosting Director. Refer to the Microsoft IIS documentation for instructions. Director configuration is not required to enable SSL.

The Director website is located at https or http://<Server_FQDN>/Director.

If one of the Sites in a multisite deployment is down, the logon for Director takes a little longer while it attempts to connect to the Site that is down.
Delegated Administration and Director

Jun 18, 2013

Delegated Administration uses three concepts: administrators, roles, and scopes. Permissions are based on an administrator's role and the scope of this role. For example, an administrator might be assigned a Help Desk administrator role where the scope involves responsibility for end-users at one site only.

For information about creating delegated administrators, see the main Delegated Administration topic.

Administrative permissions determine the Director interface presented to administrators and the tasks they can perform. Permissions determine:
- The views the administrator can access, collectively referred to as a view.
- The desktops, machines, and sessions that the administrator can view and interact with.
- The commands the administrator can perform, such as shadowing a user's session or enabling maintenance mode.

The built-in roles and permissions also determine how administrators use Director:

<table>
<thead>
<tr>
<th>Administrator Role</th>
<th>Permissions in Director</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Administrator</td>
<td>Full access to all views and can perform all commands, including shadowing a user's session, enabling maintenance mode, and exporting trends data.</td>
</tr>
<tr>
<td>Delivery Group Administrator</td>
<td>Full access to all views and can perform all commands, including shadowing a user's session, enabling maintenance mode, and exporting trends data.</td>
</tr>
<tr>
<td>Read Only Administrator</td>
<td>Can access all views and see all objects in specified scopes as well as global information. Can download reports from HDX channels and can export Trends data using the Export option in the Trends view. Cannot perform any other commands or change anything in the views.</td>
</tr>
<tr>
<td>Help Desk Administrator</td>
<td>Can access only the Help Desk and User Details views and can view only objects that the administrator is delegated to manage. Can shadow a user's session and perform commands for that user. Can perform maintenance mode operations. Can use power control options for Desktop OS Machines. Cannot access the Dashboard, Trends, or Filters views. Cannot use power control options for Server OS machines.</td>
</tr>
<tr>
<td>Machine Catalog Administrator</td>
<td>No access. This administrator is not supported for Director and cannot view data.</td>
</tr>
<tr>
<td>Host Administrator</td>
<td>No access. This administrator is not supported for Director and cannot view data.</td>
</tr>
</tbody>
</table>

Note: You can modify the views in Director, such as removing or enabling buttons, through Delegated Administrator roles and permissions. For example, if your organization does not want a certain group of Director administrators to shadow
users, select a role or customize a role that does not allow shadowing or does not allow the custom permission "Perform Remote Assistance on a machine," and the Shadow button will not appear in the UI.

To configure custom roles for Director administrators

In Studio, you can also configure Director-specific, custom roles to more closely match the requirements of your organization and delegate permissions more flexibly. For example, you can restrict the built-in Help Desk administrator role so that this administrator cannot log off sessions.

If you create a custom role with Director permissions, you must also give that role other generic permissions:
- Delivery Controller permission to log on to Director.
- Permissions to Delivery Groups to view the data related to those Delivery Groups in Director.

Alternatively, you can create a custom role by copying an existing role and include additional permissions for different views. For example, you can copy the Help Desk role and include permissions to view the Dashboard or Filters view.

Select the Director permissions for the custom role, including:

- Perform Kill Application running on a machine
- Perform Kill Process running on a machine
- Perform Remote Assistance on a machine
- Perform Reset vDisk operation
- Reset user profiles
- View Dashboard page
- View Help Desk page
- View Host Connection status and alerts
- View Slice and Dice page (Filters view)
- View Trends
- View User Details page

In addition, from the list of permissions for other components, consider these permissions:

- From Profile management: Read Profile Definitions
- From Delivery Groups:
  - Enable/disable maintenance mode of a machine using Delivery Group membership
  - Perform power operations on Windows Desktop machines using Delivery Group membership
  - Perform session management on machines using Delivery Group membership
  - View Delivery Groups
Configure HDX Insight

Jun 19, 2013

Note: The availability of this feature depends on your organization's license and your administrator permissions.

HDX Insight is the integration of EdgeSight network analysis and EdgeSight performance management with Director:

- EdgeSight network analysis leverages HDX Insight to provide an application and desktop contextual view of the network. With this feature, Director provides advanced analytics of ICA traffic in their XenDesktop deployment.
- EdgeSight performance management provides the historical retention and trend reporting. With historical retention of data versus the real-time assessment, you can create Trend reports, including capacity and health trending.

After you enable this feature in Director, HDX Insight provides Director with additional information:

- The Trends page shows latency and bandwidth effects for applications, desktops, and users across the entire deployment.
- The User Details page shows latency and bandwidth information specific to a particular user session.

Limitations

- ICA session Round Trip Time (RTT) shows data correctly for Receiver for Windows 3.4 or higher and the Receiver for Mac 11.8 or higher. For earlier versions of these Receivers, the data does not display correctly.
- In the Trends view, HDX connection logon data is not collected for VDAs earlier than 7. For earlier VDAs, the chart data is displayed as 0.

To configure the EdgeSight network analysis feature on Director

EdgeSight provides network analysis by leveraging NetScaler HDX Insight to provide the Citrix application and desktop administrators the ability to troubleshoot and correlate issues that can be attributed to poor network performance.

NetScaler Insight Center must be installed and configured in Director to enable EdgeSight network analysis. Insight Center is a virtual machine (appliance) downloaded from Citrix.com. Using EdgeSight network analysis, Director communicates and gathers the information that is related to your XenDesktop deployment. This information is leveraged from HDX Insight, which provides robust analysis of the Citrix ICA protocol between the client and the back-end Citrix infrastructure.

1. On the server where Director is installed, locate the DirectorConfig command line tool in C:\inetpub\wwwroot\Director\tools, and run it with parameter /confignetscaler in command line prompt.
2. When prompted, configure the NetScaler Insight Center machine name (FQDN or IP address), username, password, and HTTP or HTTPS connection type.
3. To verify the changes, log off and log back on.
Configure permissions for VDAs earlier than XenDesktop 7

Jun 19, 2013

If users have VDAs earlier than XenDesktop 7 installed on their devices, Director supplements information from the deployment with real-time status and metrics through Windows Remote Management (WinRM).

In addition, use this procedure to configure WinRM for use with Remote PC in XenDesktop 5.6 Feature Pack1.

By default, only local administrators of the desktop machine (typically domain administrators and other privileged users) have the necessary permissions to view the real-time data.

For information about installing and configuring WinRM, see CTX125243.

To enable other users to view the real-time data, you must grant them permissions. For example, suppose there are several Director users (HelpDeskUserA, HelpDeskUserB, and so on) who are members of an Active Directory security group called HelpDeskUsers. The group has been assigned the Help Desk administrator role in Studio, providing them with the required Delivery Controller permissions. However, the group also needs access to the information from the desktop machine.

To provide the necessary access, you can configure the required permissions in one of two ways:

- Grant permissions to the Director users (impersonation model)
- Grant permissions to the Director service (trusted subsystem model)

To grant permissions to the Director users (impersonation model)

By default, Director uses an impersonation model: The WinRM connection to the desktop machine is made using the Director user's identity. It is therefore the user that must have the appropriate permissions on the desktop.

You can configure these permissions in one of two ways (described later in this topic):

1. Add users to the local Administrators group on the desktop machine.
2. Grant users the specific permissions required by Director. This option avoids giving the Director users (for example, the HelpDeskUsers group) full administrative permissions on the machine.

To grant permissions to the Director service (trusted subsystem model)

Instead of providing the Director users with permissions on the desktop machines, you can configure Director to make WinRM connections using a service identity and grant only that service identity the appropriate permissions.

With this model, the users of Director have no permissions to make WinRM calls themselves. They can only access the data using Director.

The Director application pool in IIS is configured to run as the service identity. By default, this is the APPPOOL\Director virtual account. When making remote connections, this account appears as the server's Active Directory computer account; for example, MyDomain\DirectorServer$. You must configure this account with the appropriate permissions.

If multiple Director websites are deployed, you must place each web server's computer account into an Active Directory security group that is configured with the appropriate permissions.

To set Director to use the service identity for WinRM instead of the user's identity, configure the following setting, as
described in **Advanced Configuration:**

Service.Connector.WinRM.Identity = Service

You can configure these permissions in one of two ways:

1. Add the service account to the local Administrators group on the desktop machine.
2. Grant the service account the specific permissions required by Director (described next). This option avoids giving the service account full administrative permissions on the machine.

**To assign permissions to a specific user or group**

The following permissions are required for Director to access the information it requires from the desktop machine through WinRM:

- Read and execute permissions in the WinRM RootSDDL
- WMI namespace permissions:
  - root/cimv2 — remote access
  - root/citrix — remote access
  - root/RSOP — remote access and execute
- Membership of these local groups:
  - Performance Monitor Users
  - Event Log Readers

The ConfigRemoteMgmt.exe tool, used to automatically grant these permissions, is on the XenDesktop installation media in the x86\Virtual Desktop Agent and x64\Virtual Desktop Agent folders and on the installation media in the tools folder. You must grant permissions to all Director users.

To grant the permissions to an Active Directory security group, user, or computer account, run the tool with administrative privileges from a command prompt using the following arguments:

`ConfigRemoteMgmt.exe /configwinrmuser domain\name`

where name is a security group, user, or computer account.

For example, to grant the required permissions to a user security group:

`ConfigRemoteMgmt.exe /configwinrmuser MyDomain\HelpDeskUsers`

Or to grant the permissions to a specific computer account:

`ConfigRemoteMgmt.exe /configwinrmuser MyDomain\DirectorServer$`
Advanced configuration

Jun 19, 2013

Some advanced Director configuration, such as supporting multiple sites or multiple Active Directory forests, is controlled through settings in Internet Information Services (IIS) Manager.

Important: When you change a setting in IIS, the Director service automatically restarts and logs off users.

To configure advanced settings using IIS:

1. Open the Internet Information Services (IIS) Manager console.
2. Go to the Director website under the Default website.
3. Double-click Application Settings.
4. Double-click a setting to edit it.

Director uses Active Directory to search for users and to look up additional user and machine information. By default, Director searches the domain or forest in which:
- The administrator's account is a member.
- The Director web server is a member (if different).

Director attempts to perform searches at the forest level using the Active Directory global catalog. If the administrator does not have permissions to search at the forest level, only the domain is searched.

To search or look up data from another Active Directory domain or forest requires that you explicitly set the domains or forests to be searched. Configure the following setting:

`Connector.ActiveDirectory.Domains = (user),(server)`

The value attributes user and server represent the domains of the Director user (the administrator) and Director server respectively.

To enable searches from an additional domain or forest, add the name of the domain to the list, as shown in this example:

`Connector.ActiveDirectory.Domains = (user),(server),ENDUSERDOMAIN`

For each domain in the list, Director attempts to perform searches at the forest level. If the administrator does not have permissions to search at the forest level, only the domain is searched.

If Director is already installed, configure it to work with multiple sites. To do this, use the IIS Manager Console on each Director server to update the list of server addresses in the application settings.

Add an address of a controller from each site to the following setting:

`Service.AutoDiscoveryAddresses = SiteAController,SiteBController`

where SiteAController and SiteBController are the addresses of Delivery Controllers from two different sites.

By default, the Activity Manager in Director displays a list of all the running applications and the Windows description in the title bars of any open applications for the user's session. This information can be viewed by all administrators that have access to the Activity Manager feature in Director. For Delegated Administrator roles, this includes Full administrator,
Delivery Group administrator, and Help Desk Administrator.

To protect the privacy of users and the applications they are running, you can disable the Applications tab from listing running applications.

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

1. On the VDA, modify the registry key located at HKLM\Software\Citrix\Director\TaskManagerDataDisplayed. By default, the key is set to 1. Change the value to 0, which means the information will not be displayed in the Activity Manager.

2. On the server with Director installed, modify the setting that controls the visibility of running applications. By default, the value is true, which allows visibility of running applications in the Applications tab. Change the value to false, which disables visibility. This option affects only the Activity Manager in Director, not the VDA.

Modify the value of the following setting:

UI.TaskManager.EnableApplications = false

Important: To disable the view of running applications, Citrix recommends making both changes to ensure the data is not displayed in Activity Manager.
Troubleshoot user issues

Jun 10, 2013

Use the Director's Activity Manager to view information about the user:

- Check for details about the user's logon, connection, and applications.
- Shadow the user's machine.
- Troubleshoot the issue with the recommended actions in the following table, and, if needed, escalate the issue to the appropriate administrator.

### Troubleshooting tips

<table>
<thead>
<tr>
<th>User's issue</th>
<th>See these suggestions:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logon takes a long time or fails intermittently or repeatedly</td>
<td>Diagnose user logon issues</td>
</tr>
<tr>
<td>Application is slow or won't respond</td>
<td>Resolve application failures</td>
</tr>
<tr>
<td>Connection failed</td>
<td>Restore desktop connections</td>
</tr>
<tr>
<td>Session is slow or not responding</td>
<td>Restore sessions</td>
</tr>
<tr>
<td>Video is slow or poor quality</td>
<td>Run HDX channel system reports</td>
</tr>
</tbody>
</table>

Note: To make sure that the machine is not in maintenance mode, from the User Details view, review the Machine Details panel.
In the User Details view, the HDX panel does not display mapped client drives, and mapped printers are not detected in the printing channel.
Upload troubleshooting information

Sep 26, 2013
Run Citrix Scout from a single Delivery Controller or VDA to capture key data points and Citrix Diagnosis Facility (CDF) traces to troubleshoot selected computers. After capturing this information, Scout securely uploads the data points to Citrix Technical Support. The Tools As a Service (TaaS) platform uses this information to reduce the time to resolve customer-reported issues.

This release of Scout is installed with other XenDesktop components. Scout appears in the Windows Start Menu or Windows 8 or 8.1 Start Screen when you install or upgrade to XenDesktop 7.1.

To start Scout, from the Start Menu or Start Screen, select Citrix > Citrix Scout.

For information on using and configuring Scout, and for frequently asked questions, see http://support.citrix.com/article/CTX130147.

The following video summarizes how to use Scout.
Search tips

May 10, 2013

When you type the user's name in a Search field, Director searches for users in Active Directory for users across all sites configured to support Director.

The search results also include users who are not currently using or assigned to a machine.

- Searches are not case-sensitive.
- Partial entries produce a list of possible matches.
- After you type a few letters of a two-part name (username, family name and first name, or display name), separated by a space, the results include matches for both strings. For example, if you type jo rob, the results might include strings such as “John Robertson” or Robert, Jones.

To return to the landing page, click the Director logo.
Shadow users

Jun 21, 2013

From Director, use the shadow user feature to view and work directly on a user's virtual machine or session. The user must be connected to the machine that you want to shadow. Verify this by checking the machine name listed in the user title bar.

1. In the User Details view, select the user session.
2. Activate shadowing for the selected user session:
   - For machine monitoring, in the Activity Manager view, click Shadow.
   - For session monitoring, in the User Details view, locate the Session Details panel and click Shadow.
3. After the connection initializes, a dialog box prompts you to open or save the .msrcincident file.
4. Open the incident file with the Remote Assistance Viewer, if not already selected by default. A confirmation prompt appears on the user device.
5. Instruct the user to click Yes to start the machine or session sharing.

For additional control, ask the user to share keyboard and mouse control.

Configure your Microsoft Internet Explorer browser to automatically open the downloaded Microsoft Remote Assistance (.msra) file with the Remote Assistance client.

To do this, you must enable one security setting in your browser: DownloadsAutomatic prompting for file downloads. By default, this option is enabled for sites in the Local intranet zone. If the Director site is not in the Local intranet zone, consider manually adding the site to this zone.
Send messages to users

Jun 21, 2013
From Director, send a message to a user who is connected to one or more machines. For example, use this feature to send immediate notices about administrative actions such as impending desktop maintenance, machine log-offs and restarts, and profile resets.

1. In the Activity Manager view, select the user and click Details.
2. In the User Details view, locate the Session Details panel and click Send Message.
3. Type your message information in the Subject and Message fields, and click Send.

If the message is sent successfully, a confirmation message appears in Director. If the user's machine is connected, the message appears there.

If the message is not sent successfully, an error message appears in Director. Troubleshoot the problem according to the error message. When you have finished, type the subject and message text again and click Try again.
Diagnose user logon issues

Jun 03, 2013

Use these general steps:

1. From the User Details view, troubleshoot the logon state using the Logon Duration panel.
   - If the user is logging on, the view reflects the process of logging on.
   - If the user is currently logged on, the Logon Duration panel displays the time it took for the user to log on to the current session.

2. Ask the user to log off and then log on again so that you can observe the Logon Duration data. The panel typically updates after about 3 minutes, but it could take longer depending on the time taken for the logon to complete.

3. Examine the phases of the logon process:
   - **Brokering** — Time taken to decide which desktop to assign to the user.
   - **VM start** — Time taken to boot the desktop.
   - **HDX connection** — Time taken for HDX connection establishment, dependent on the network.
   - **GPOs** — Time taken to apply group policy objects.
   - **Login scripts** — Time taken for scripts.
   - **Profile load** — Time taken to load the user profile.
   - **Interactive session** — Time taken to establish an interactive user session.

   The total logon time is not an exact sum of these phases. For example, some phases occur in parallel, and in some phases, additional processing occurs that might result in a longer logon duration than the sum.

   **Tip:** To identify unusual or unexpected values in the graph, compare the amount of time taken in each phase of the current session with the average duration for this user for the last seven days, and the average duration for all users in this Delivery Group for the last seven days.

   Escalate as needed. For example, if the VM startup is slow, the issue could be in the hypervisor, so you can escalate it to the hypervisor administrator. Or, if the brokering time is slow, you can escalate the issue to the XenDesktop administrator to check the load balancing on the Delivery Controller.

**Troubleshooting tips:** Examine unusual differences, including:

- **Missing (current) logon bars**
- **Major discrepancy between the current duration and this user's average duration.** Causes could including:
  - A new application was installed.
  - An operating system update occurred.
  - Configuration changes were made.
- **Major discrepancy between the user's logon numbers (current and average duration) and the Delivery Group average duration.**

If needed, click Restart to observe the user's logon process to troubleshoot issues, such as VM Start or Brokering.
Resolve application failures

Jun 19, 2013

In the Activity Manager view, click the Applications tab. You can view all the applications on all machines to which this user has access, including local and hosted applications for the currently connected machine, and the current status of each. Note: If the Applications tab is greyed out, contact an administrator with the permission to enable the tab. The list includes only those applications that were launched within the session.

For Server OS machines and Desktop OS machines, applications are listed for each disconnected session. If the user is not connected, no applications are displayed.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>End the application that is not responding.</td>
<td>Choose the application that is not responding and click End Application. Once the application is terminated, ask the user to launch it again.</td>
</tr>
<tr>
<td>End processes that are not responding.</td>
<td>If you have the required permission, click the Processes tab. Select a process that is related to the application or using a high amount of CPU resources or memory, and click End Process. However, if you do not have the required permission to terminate the process, attempting to end a process will fail.</td>
</tr>
<tr>
<td>Restart the user's machine.</td>
<td>For Desktop OS machines only, for the selected session, click Restart. Alternatively, from the Machine Details view, use the power controls to restart or shut down the machine. Instruct the user to log on again so that you can recheck the application. For Server OS machines, the restart option is not available. Instead, log off the user and let the user log on again.</td>
</tr>
<tr>
<td>Put the machine into maintenance mode.</td>
<td>If the machine's image needs maintenance, such as a patch or other updates, put the machine into maintenance mode and escalate the issue to the appropriate administrator. Click , and from the Machine Details view, click Details and turn on the maintenance mode option. Escalate to the appropriate administrator.</td>
</tr>
</tbody>
</table>
Restore desktop connections

Jun 03, 2013

From Director, check the user's connection status for the current machine in the user title bar.

If the desktop connection failed, the error that caused failure is displayed and can help you decide how to troubleshoot.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensure that the machine is not in maintenance mode.</td>
<td>On the User Details page, make sure maintenance mode is turned off.</td>
</tr>
<tr>
<td>Restart the user's machine.</td>
<td>Select the machine and click Restart. Use this option if the user's machine is unresponsive or unable to connect, such as when the machine is using an unusually high amount of CPU resources, which can make the CPU unusable.</td>
</tr>
</tbody>
</table>
Restore sessions

Jun 19, 2013

If a session becomes disconnected, it is still active and its applications continue to run, but the user device is no longer communicating with the server.

In the User Details view, troubleshoot session failures in the Session Details panel. You can view the details of the current session, indicated by the session ID.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>End applications or processes</td>
<td>Click the Applications tab. Select any application that is not responding and click End Application. Similarly, select any corresponding process that is not responding and click End Process. Also, end processes that are consuming an unusually high amount of memory or CPU resources, which can make the CPU unusable.</td>
</tr>
<tr>
<td>that are not responding.</td>
<td></td>
</tr>
<tr>
<td>Disconnect the Windows session.</td>
<td>Click Session Control and then select Disconnect. This option is available only for brokered Server OS machines. For non-brokered sessions, the option is disabled.</td>
</tr>
<tr>
<td>Log off the user from the session.</td>
<td>Click Session Control and then select Log Off.</td>
</tr>
</tbody>
</table>

To test the session, the user can attempt to log back onto it. You can also shadow the user to more closely monitor this session.

Note: If user devices are running Virtual Delivery Agents (VDAs) earlier than XenDesktop 7, Director cannot display complete information about the session; instead, it displays a message that the information is not available. These messages might appear in the User Details page and Activity Manager.
Run HDX channel system reports

May 14, 2013

In the User Details view, check the status of the HDX channels on the user’s machine in the HDX panel. This panel is available only if the user machine is connected using HDX.

If a message appears indicating that the information is not currently available, wait for one minute for the page to refresh, or select the Refresh button. HDX data takes a little longer to update than other data.

Click an error or warning icon for more information.

Tip: You can view information about other channels in the same dialog box by clicking the left and right arrows in the left corner of the title bar.

HDX channel system reports are used mainly by Citrix Support to troubleshoot further.

1. In the HDX panel, click Download System Report.
2. You can view or save the .xml report file.
   - To view the .xml file, click Open. The .xml file appears in the same window as the Director application.
   - To save the .xml file, click Save. The Save As window appears, prompting you for a location on the Director machine to download the file to.
Reset a Personal vDisk

Jun 19, 2013

Caution: When you reset the disk, the settings revert back to their factory default values and all data on it is deleted, including applications. The profile data is retained unless you modified the Personal vDisk default (of redirecting profiles from the C: drive), or you are not using a third-party profile solution.

To reset, the machine with the Personal vDisk must be running; however, the user does not have to be logged on to it.

This option is available only for Desktop OS machines; it is disabled for Server OS machines.

1. From the Help Desk view, choose the targeted Desktop OS machine.
2. From this view or in the Personalization panel of the User Details view, click Reset Personal vDisk.
3. Click Reset. A message appears warning that the user will be logged off. After the user is logged off (if the user was logged on), the machine restarts.

If the reset is successful, the Personal vDisk status field value in the Personalization panel of the User Details view is Running. If the reset is unsuccessful, a red X to the right of the Running value appears. When you point to this X, information about the failure appears.
Reset a user profile

Jun 19, 2013
Caution: When a profile is reset, although the user's folders and files are saved and copied to the new profile, most user profile data are deleted (for example, the registry is reset and application settings might be deleted).

1. From Director, search for the user whose profile you want to reset and select this user's session.

2. Click **Reset Profile**.

3. Instruct the user to log off from all sessions.

4. Instruct the user to log on again. The folders and files that were saved from the user's profile are copied to the new profile.

   Important: If the user has profiles on multiple platforms (such as Windows 8 and Windows 7), instruct the user to log back on first to the same desktop or app that the user reported as a problem. This ensures that the correct profile is reset.

   If the profile is a Citrix user profile, the profile is already reset by the time the user's desktop appears. If the profile is a Microsoft roaming profile, the folder restoration might still be in progress for a brief time. The user must stay logged on until the restoration is complete.

If the profile is not successfully reset (for example, the user cannot successfully log back on to the machine or some of the files are missing), you must manually restore the original profile.

The folders (and their files) from the user's profile are saved and copied to the new profile. They are copied in the listed order:

- Desktop
- Cookies
- Favorites
- Documents
- Pictures
- Music
- Videos

Any Citrix user profile or Microsoft roaming profile can be reset. After the user logs off and you select the reset command (either in Director or using the PowerShell SDK), Director first identifies the user profile in use and issues an appropriate reset command. Director receives the information through Profile management, including information about the profile size, type, and logon timings.

The next time the user logs on, this diagram illustrates the processing that occurs.
1. The reset command issued by Director specifies the profile type. The Profile management service then attempts to reset a profile of that type and looks for the appropriate network share (user store). If the user is processed by Profile management, but receives a roaming profile command, it is rejected (or vice versa).

2. If a local profile is present, it is deleted.

3. The network profile is renamed.

4. The next action depends on whether the profile being reset is a Citrix user profile or a Microsoft roaming profile.
   - For Citrix user profiles, the new profile is created using the Profile management import rules, and the folders are copied back to the network profile, and the user can log on proceeds as normal. If a roaming profile is used for the reset, any registry settings in the roaming profile are preserved in the reset profile.
     Note: You can configure Profile management so that a template profile overrides the roaming profile, if required.
   - For Microsoft roaming profiles, a new profile is created by Windows, and when the user logs on, the folders are copied back to the user device. When the user logs off again, the new profile is copied to the network store.

1. Instruct the user to log off from all sessions.
2. Delete the local profile if one exists.
3. Locate the archived folder on the network share that contains the date and time appended to the folder name, the folder with a .upm_datestamp extension.
4. Delete the current profile name; that is, the one without the .upm_datestamp extension.
5. Rename the archived folder using the original profile name; that is, remove the date and time extension. You have returned the profile to its original, pre-reset state.
Control user machine power states

Jun 19, 2013

To control the state of the machines that you select in Director, use the Power Control options. These options are available for Desktop OS machines, but might not be available for Server OS machines.

Note: This functionality is not available for physical machines or machines using Remote PC Access.

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restart</td>
<td>Performs an orderly (soft) shutdown of the VM; all running processes are halted individually before restarting the VM. For example, select machines that appear in Director as “failed to start,” and use this command to restart them.</td>
</tr>
<tr>
<td>Force Restart</td>
<td>Restarts the VM without first performing any shut-down procedure. This command works in the same way as unplugging a physical server and then plugging it back in and turning it back on.</td>
</tr>
<tr>
<td>Shut Down</td>
<td>Performs an orderly (soft) shutdown of the VM; all running processes are halted individually.</td>
</tr>
<tr>
<td>Force Shutdown</td>
<td>Shuts down the VM without first performing any shut-down procedure. This command works in the same way as unplugging a physical server. It may not always shut down all running processes, and you risk losing data if you shut down a VM in this way.</td>
</tr>
<tr>
<td>Suspend</td>
<td>Suspends a running VM in its current state and stores that state in a file on the default storage repository. This option allows you to shut down the VM’s host server and later, after rebooting it, resume the VM, returning it to its original running state.</td>
</tr>
<tr>
<td>Resume</td>
<td>Resumes a suspended VM and restores its original running state.</td>
</tr>
<tr>
<td>Start</td>
<td>Starts a VM when it is off (also called a cold start).</td>
</tr>
</tbody>
</table>

If power control actions fail, hover the mouse over the alert, and a pop-up message appears with details about the failure.
Monitor deployments

May 20, 2013

With full administrator permissions, when you open Director, the Dashboard provides a centralized location to monitor the health and usage of a site.

If there are currently no failures and no failures have occurred in the past 60 minutes, panels stay collapsed. When there are failures, the specific failure panel automatically appears.

Note: Depending on your organization's license and your Administrator privileges, some options or features might not be available.

<table>
<thead>
<tr>
<th>Panel</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Connection Failures</td>
<td>Connection failures over the last 60 minutes. Click the categories next to the total number to view metrics for that type of failure. In the adjacent table, that number is broken out by Delivery Groups.</td>
</tr>
<tr>
<td>Failed Desktop OS Machines or Failed Server OS Machines</td>
<td>Total failures in the last 60 minutes broken out by Delivery Groups. Failures broken out by types, including failed to start, stuck on boot, and unregistered. For Server OS machines, failures also include machines reaching maximum load.</td>
</tr>
<tr>
<td>Sessions Connected</td>
<td>Connected sessions across all Delivery Groups for the last 60 minutes.</td>
</tr>
<tr>
<td>Average Logon Duration</td>
<td>Logon data for the last 60 minutes. The large number on the left is the average logon duration across the hour. Logon data for VDAs earlier than XenDesktop 7.0 is not included in this average.</td>
</tr>
<tr>
<td>Infrastructure</td>
<td>Health status of your site's hosts, controllers, and infrastructure. View performance alerts. For hosts, the connection status and the health of the CPU, memory, bandwidth (network usage), and storage (disk usage) are monitored using information from XenServer or VMware. For example, you can configure XenCenter to generate performance alerts when CPU, network I/O or disk I/O usage go over a specified threshold on a managed server or virtual machine. By default, the alert repeat interval is 60 minutes, but you can configure this as well. For details, in the XenServer documentation, see Configuring Performance Alerts.</td>
</tr>
</tbody>
</table>

Note: If no icon appears for a particular metric, this indicates that this metric is not supported by the type of host you are using. For example, no health information is available for System Center Virtual Machine Manager (SCVMM) hosts.

Continue to troubleshoot issues using these options:

- Control user machine power
- Prevent connections to machines
Monitor sessions

Jun 19, 2013

If a session becomes disconnected, it is still active and its applications continue to run, but the user device is no longer communicating with the server.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>View a user's currently connected machine or session</td>
<td>From the Activity Manager and User Details views, view the user's currently connected machine or session and a list of all machines and sessions to which this user has access. To access this list, click the session switcher icon in the user title bar. See Restore sessions.</td>
</tr>
<tr>
<td>View the total number of connected sessions across all Delivery Groups</td>
<td>From the Dashboard, in the Sessions Connected pane, view the total number of connected sessions across all Delivery Groups for the last 60 minutes. Then click the large total number, which opens the Filters view, where you can display graphical session data based on selected Delivery Groups and ranges and usage across Delivery Groups. See Filter data to troubleshoot failures.</td>
</tr>
<tr>
<td>View data over a longer period of time</td>
<td>On the Trends view, select the Sessions tab to drill down to more specific usage data for connected and disconnected sessions over a longer period of time (that is, session totals from earlier than the last 60 minutes). To view this information, click View historical trends. See Monitor historical trends across a site.</td>
</tr>
</tbody>
</table>

Note: If the user device is running a legacy Virtual Delivery Agents (VDA), such as a VDA earlier than version 7, Director cannot display complete information about the session. Instead, it displays a message that the information is not available in the User Details view and Activity Manager panel.
Create filters

May 20, 2013

When you click numbers on the Dashboard or select a predefined filter from the Filter menu, the Filter view opens to display the data based on the selected machine or failure type.

Predefined filters cannot be edited, but you can save a predefined filter as a custom filter and then modify it. Additionally, you can create custom filtered views of machines, connections, and sessions across all Delivery Groups.

1. Select a view:
   - Machines — Select Desktop OS Machines or Server OS Machines. These views show the number of configured machines. The Server OS Machines tab also includes the load evaluator index, which indicates the distribution of performance counters and tool tips of the session count if you hover over the link.
   - Sessions — You can also see the session count from the Machines view.
   - Connections — Filter connections by different time periods, including last 60 minutes, last 24 hours, or last 7 days. For each view, you can use the preselected columns, or add additional columns to include more information.

2. For Failure by, select the criteria.

3. Use the additional tabs for each view, as needed, to complete the filter.

4. Select additional columns, as needed, to troubleshoot further.

5. Save and name your filter. To open filter later, from the Filter menu, select the failure type (Machines, Sessions, or Connections), and then select the saved filter.

6. If needed, for Machines or Connections views, use power controls for all the machines you select in the filtered list. For the Sessions view, use the session controls or option to send messages.

From the filtered results, continue to troubleshoot issues using these options:

- Control user machine power
- Prevent connections to machines
Monitor historical trends across a site

Sep 04, 2013

The Trends view accesses historical trend information for sessions, connection failures, machine failures, logon performance, and load evaluation for each site. To locate this information, from the Dashboard or Filters view, click Trends.

To change the default scope of each graph, apply a different filter to the data. See Filter data to troubleshoot failures for details about how to select parameters on this view.

Action | Description
---|---
Export graph data | Select the tab containing the data to export. Click Export and select the file format: .PDF or .CVS.
View trends for sessions | From the Sessions tab, select the Delivery Group and time period to view more detailed information about the concurrent session count.
View trends for connection failures | From the Connection Failures tab, select the machine type, failure type, Delivery Group, and time period to view a graph containing more detailed information about the user connection failures across your site.
View trends for machine failures | From the Desktop OS Machine Failures tab or Server OS Machines tab, select the failure type, Delivery Group, and time period to view a graph containing more detailed information about the machine failures across your site.
View trends for logon performance | From the Logon Performance tab, select the Delivery Group and time period to view a graph containing more detailed information about the duration of user logon times across your site and whether the number of logons affects the performance. This view also shows the average duration of the logon phases, such as brokering duration, VM start time, and so on. This data is specifically for user logons and does not include users trying to reconnect from disconnected sessions.
View trends for load evaluation | From the Load Evaluator Index tab, view a graph containing more detailed information about the load that is distributed among Server OS machines. The filter options for this graph include the Delivery Group or Server OS machine (available only if Server OS machine in a Delivery Group was selected), and range.
View network analysis data using HDX Insight | The availability of this feature depends on your organization's license and your administrator permissions. From the Network tab, monitor your network analysis, which provides a user, application, and desktop contextual view of the network. With this feature, Director provides advanced analytics of ICA traffic in your deployment.

The flag icons on the graph indicate significant events or actions for that specific time range. Hover the mouse over the flag and click to list events or actions.
Note: HDX connection logon data is not collected for VDAs earlier than 7. For earlier VDAs, the chart data is displayed as 0. Continue to troubleshoot issues using these options:

- Control user machine power
- Prevent connections to machines
Prevent connections to machines

May 21, 2013

Use maintenance mode to prevent new connections temporarily while the appropriate administrator performs maintenance tasks on the image.

When you enable maintenance mode on machines, no new connections are allowed until you disable it. If users are currently logged on, maintenance mode takes effect as soon as all users are logged off. For users who do not log off, send a message informing them that machines will be shut down at a certain time, and use the power controls to force the machines to shut down.

1. Select the machine, such as from the User Details view, or a group of machines in the Filters view.
2. Select Maintenance Mode, and turn on the option.

If a user tries to connect to an assigned desktop while it is in maintenance mode, a message appears indicating that the desktop is currently unavailable. No new connections can be made until you disable maintenance mode.
You can use a diagnostic tool to monitor the changes made by users to both parts of their Personal vDisks (the user data and the application parts). These changes include applications that users have installed and files they have modified. The changes are stored in a set of reports.

1. On the machine that you want to monitor, run C:\Program Files\Citrix\personal vDisk\bin\CtxPvdDiag.exe.
2. Browse to a location where you want to store the reports and logs, select which reports to generate, and click OK. The following reports are available:

<table>
<thead>
<tr>
<th>Report or Log</th>
<th>Generated Files</th>
<th>Changes Monitored</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software hive report</td>
<td>Software.Dat.Report.txt, Software.Dat.delta.txt</td>
<td>Software.Dat.Report.txt records the changes made by the user to the HKEY_LOCAL_MACHINE\Software hive. It consists of the following sections:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● List of Applications installed on the base — The applications that were installed in Layer 0.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● List of user installed software — the applications that were installed by the user on the application part of the vDisk.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● List of software uninstalled by user — the applications removed by the user that were originally present in Layer 0.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See Hive delta report for information on Software.Dat.delta.txt.</td>
</tr>
<tr>
<td>System hive report</td>
<td>SYSTEM.CurrentControlSet.DAT.Report.txt</td>
<td>This file records the changes made by the user to the HKEY_LOCAL_MACHINE\System hive. It contains the following sections:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● List of user installed services — the services and drivers installed by the user.</td>
</tr>
<tr>
<td>Security hive report</td>
<td>SECURITY.DAT.Report.txt</td>
<td>This file monitors all changes that the user makes in the HKEY_LOCAL_MACHINE\Security hive.</td>
</tr>
<tr>
<td>Security Account Manager(SAM) hive report</td>
<td>SAM.DAT.Report.txt</td>
<td>This file monitors all changes that the user makes in the HKEY_LOCAL_MACHINE\SAM hive.</td>
</tr>
<tr>
<td>Hive delta</td>
<td>Software.Dat.delta.txt</td>
<td>This file records all registry keys and values added or...</td>
</tr>
<tr>
<td>Report or Log</td>
<td>Generated Files</td>
<td>Changes Monitored</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Personal vDisk logs</td>
<td>Pvd-IvmSupervisor.log, PvdActivation.log, PvdSvc.log, PvdWM1.log, SysVol-IvmSupervisor.log, vDeskService-&lt;#&gt;.log</td>
<td>These files are generated by default in P:\Users&lt;user account&gt;\AppData\Local\Temp\PVDLOGS but are moved to the selected location.</td>
</tr>
<tr>
<td>Windows operating system (OS) log</td>
<td>EvtLog_App.xml, EvtLog_System.xml, setupapi.app.log, setuperr.log, setupapi.dev.log, msinfo.txt</td>
<td>EvtLog_App.xml and EvtLog_System.xml are the application and system event logs in XML format from the Personal vDisk volume. Setupapi.app.log and setuperr.log contain log messages from when msiexec.exe was run during Personal vDisk installation. Setupapi.dev.log contains device installation log messages. Msinfo.txt contains the output of msinfo32.exe. For information on this output, see your Microsoft documentation.</td>
</tr>
</tbody>
</table>
| File system report               | FileSystemReport.txt                                                             | This file records changes made by the user to the file system. It consists of the following sections:  
  * Files Relocated — the files in Layer 0 that were moved by the user to the vDisk. Layer 0 files are those that were inherited from the master image by the machine to which the Personal vDisk is attached.  
  * Files Removed — the files in Layer 0 that were hidden by a user's action (for example, removing an application).  
  * Files Added (MOF,INF,SYS) — the files with .mof, .inf, or .sys extensions that the user added to the vDisk (for example, when they installed an application such as Visual Studio 2010 that registers a .mof file for autorecovery).  
  * Files Added Other — Other files that the user added to the vDisk (for example, when they installed an application).  
  * Base Files Modified But Not Relocated — the files in Layer 0 that the user modified but that the Personal vDisk Kernel-Mode drivers did not capture in the vDisk. |
These topics include information about the XenDesktop SDK and the XenDesktop Policy Settings Reference. Also included is information about using the Monitor service application programming interface (API).

<table>
<thead>
<tr>
<th>XenDesktop SDK</th>
<th>Policy settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitor Service OData API</td>
<td></td>
</tr>
</tbody>
</table>
About the XenDesktop SDK

Sep 23, 2013
This topic introduces the XenDesktop SDK and tells you how to access and use it. Key differences between the XenDesktop 5 and XenDesktop 7 SDK are also explained.

XenDesktop provides an SDK based on a number of Microsoft Windows PowerShell version 3.0 snap-ins that allows you to perform the same tasks as you would with the Citrix Studio console, together with tasks you cannot do with Studio alone.

Note: The PowerShell SDK is compatible only with SDKs associated with XenDesktop version 5.0 and later.

- New high-level SDK — XenDesktop 7 provides a new high-level SDK that enables you to script and automate site creation and maintenance quickly and easily. The high-level SDK insulates you from much of the complexity of the low-level SDKs, such that you can create a new site simply by running two cmdlets.

- New low-level SDKs — Individual low-level SDKs are provided for the new XenDesktop 7 services, including a dedicated and enhanced SDK for the Delegated Administration Service (DAS), which was previously part of the Broker SDK in XenDesktop 5. There are also SDKs for new features including the Monitor Service, Environment Test, and Configuration Logging.

- Windows Server OS Machine catalogs and delivery groups — You can use the XenDesktop 7 SDK to deliver cost-effective applications and desktops hosted on server operating systems.

- Desktop OS Machine applications — Desktop OS Machine applications have changed significantly at the SDK level. If you have existing scripts for running applications on Desktop OSs, you will have to update these scripts for XenDesktop 7 as there is little backwards compatibility.

- Apply settings to machines in Delivery Groups — In XenDesktop 7, using configuration slots, you can apply settings to machines in a specific delivery group, rather than to all machines in a site. This enables you to configure, for a given delivery group, which settings apply to that group. A number of pre-defined configuration slots are provided that contain different types of settings, such as settings for StoreFront addresses for use with Receiver or App-V publishing server locations. You can use one collection of settings from a slot to affect only a particular delivery group, and a different collection of settings from the same slot to affect another delivery group. You can use names appropriate to your particular deployment; for example, "Sales Department policy."

- Catalog types replaced — In XenDesktop 7, catalog types have been replaced by catalogs with individual properties. However, for backwards compatibility, you can still use existing scripts that employ catalog types, such as single image (pooled) and thin clone (dedicated) etc., but internally these are converted into sets of properties.

  Caution: Backwards compatibility with XenDesktop 5 catalog types has been maintained where possible and practicable. However, when writing new scripts, do not use catalog types; instead, specify catalogs with individual properties.

- Desktop object replaced — In XenDesktop 5, the Desktop object is one of the main types of SDK object used in Broker SDK scripts. The Desktop object describes both the machine and the session on the machine. In XenDesktop 7, this object is replaced by the Session object and the Machine object, both of which have been expanded to do the work of the Desktop object. However, for backwards compatibility, you can still use existing scripts that employ the Desktop object.

  Caution: Backwards compatibility with XenDesktop 5 has been maintained where possible and practicable. However, when writing new scripts, do not use the Desktop object; instead, specify Session and Machine objects.

- Terminology — The terminology used throughout the XenDesktop SDK differs in places from that in the Studio console. Therefore, you may see references to "desktop group" throughout the SDK help because the SDK does not have the same concept of a "delivery group". Similarly, the term "hypervisor" appears in the SDK help, where this is
referred to as "host" in the console.

There are differences between the XenDesktop SDK and the Studio console in terms of policy rules. Entitlement and assignment policy rules are independent entities in the SDK; in the console, these entities are not visible as they are seamlessly merged with the Delivery Group. Also, access policy rules are less restrictive in the SDK.

The XenDesktop SDK comprises of a number of PowerShell snap-ins installed automatically by the XenDesktop installation wizard when you install the Controller or Studio components.

To access and run the cmdlets:

1. Start a shell in PowerShell 3.0.
   To start a shell from the console, click Studio, select the PowerShell tab, and click on Launch PowerShell.
   You must run the shell or script using an identity that has Citrix administration rights. Although members of the local administrators group on the Controller automatically have full administrative privileges to allow XenDesktop to be installed, Citrix recommends that for normal operation, you create Citrix administrators with the appropriate rights, rather than use the local administrators account. If you are running Windows Server 2008, you must run the shell or script as a Citrix administrator, and not as a member of the local administrators group.

2. To use XenDesktop SDK cmdlets within scripts, set the execution policy in PowerShell.
   For more information about PowerShell execution policy, see your Microsoft documentation.

3. Add the snap-ins you require into the PowerShell environment using the Add-PSSnapin command in the Windows PowerShell console. V1 and V2 denote the version of the snap-in (XenDesktop 5 snap-ins are version 1; XenDesktop 7 snap-ins are version 2). For example, type:
   Add-PSSnapin Citrix.ADIdentity.Admin.V2
   To import all the XenDesktop cmdlets, type:
   Add-PSSnapin Citrix.*.Admin.V*
   After importing, you have access to the XenDesktop cmdlets and their associated help.

For an example of a typical use case, see Get started with the XenDesktop SDK.

Tip: For a complete listing of all help text for the XenDesktop cmdlets, see XenDesktop cmdlet help.

The Citrix Group Policy SDK allows you to display and configure Group Policy settings and filters. It uses a PowerShell provider to create a virtual drive that corresponds to the machine and user settings and filters. The provider appears as an extension to New-PSDrive. To use the Group Policy SDK, either Studio or the XenDesktop SDK must be installed.

Adding the Group Policy SDK

1. To add the Group Policy SDK, type:
   Add-PSSnapin citrix.common.grouppolicy

2. To access help, type:
   help New-PSDrive -path localgpo:"
Using the Group Policy SDK

1. To create a virtual drive and load it with settings, type:
   ```powershell
   New-PSDrive <Standard Parameters> [-PSProvider] CitrixGroupPolicy
   -Controller <string>
   ```

   where
   ```powershell
   -Controller
   ```
   is the fully qualified domain name of a controller in the XenDesktop site you want to connect to and load settings from.
Get started with the XenDesktop SDK

Oct 03, 2013
These topics explain how to use the SDK in your XenDesktop deployment. The recommended procedure for creating a script is described; you can use this procedure as the basis for creating scripts for other purposes. Examples of typical use cases are also provided.

Creating a script

To create a script, perform the following steps:

1. Use the Citrix Studio console to perform the operation that you want to script; for example, to create a catalog for a set of Machine Creation Services Machines.
2. Collect the log of SDK operations that Studio made to perform the task.
3. Review the script to understand what each part is doing. This will help you with the customization of your own script. For more information, see the example use case which explains in detail what the script is doing.
4. Convert and adapt the Studio script fragment to turn it into a script that is more consumable. To do this:
   - Use variables. Some cmdlets take parameters, such as TaskId. However, it may not be clear where the value used in these parameters comes from because Studio uses values from the result objects from earlier cmdlets.
   - Remove any commands that are not required.
   - Add some steps into a loop so that these can be easily controlled. For example, add machine creation into a loop so that the number of machines being created can be controlled.

Examples

Note: When creating a script, to ensure you always get the latest enhancements and fixes, Citrix recommends you follow the procedure described above rather than copying and pasting the example scripts.

<table>
<thead>
<tr>
<th>Examples</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create catalog</td>
<td>Script: create a catalog for a set of Machine Creation Services (MCS) machines</td>
</tr>
<tr>
<td>Example: Create and configure a host</td>
<td>Script: create and configure a host</td>
</tr>
<tr>
<td>Example: Create a PvD Desktop</td>
<td>Script: create a Delivery Group containing Personal vDisk (PvD) desktops</td>
</tr>
<tr>
<td>Example: Get load balancing information</td>
<td>Display load index values for Server OS Machines</td>
</tr>
</tbody>
</table>
Example: Create a catalog

Nov 04, 2014
The following example shows how to create a catalog for a set of Machine Creation Services (MCS) machines.

Before you begin, make sure you follow the steps detailed in Get started with the XenDesktop SDK. This topic shows how to use Studio to perform the operation you want to script (in this case, to create a catalog for a set of Machine Creation Services machines) and collect the log of SDK operations that Studio made to perform the task. This output can then be customized to produce a script for automating catalog creation.

Note: To ensure you always get the latest enhancements and fixes, Citrix recommends you follow the procedure described in this topic, rather than copying and pasting the example script. Line numbers and line breaks have been added to the script for readability.

Understand the script

The following section explains what each part of the script produced by Studio is doing. This will help you with the customization of your own script. Line numbers have been added for readability.

1. Start-LogHighLevelOperation -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -Source 'Studio' -StartTime 29/05/2013 14:43:08 -Text 'Create Machine Catalog ExampleMachines'

Starts a logged operation and returns a log ID which is supplied to subsequent operations to associate them with the larger task.

2. New-BrokerCatalog -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -AllocationType 'Permanent' -Description 'Example Machines' -IsRemotePC $False -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 -MinimumFunctionalLevel 'L7' -Name 'ExampleMachines' -PersistUserChanges 'OnPvd' -ProvisioningType 'MCS' -Scope @() -SessionSupport 'SingleSession'

Creates a Broker catalog. This catalog is populated with machines which are about to be created.

3. New-AcctIdentityPool -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -AllowUnicode -Domain 'dumdev.internal.citrix.com' -IdentityPoolName 'ExampleMachines' -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 -NamingScheme 'Example-####' -NamingSchemeType 'Numeric' -OU 'OU=DUM VMs,DC=dumdev,DC=internal,DC=citrix,DC=com' -Scope @()

Creates an Identity Pool. This defines the mechanism for creating AD computer accounts. This becomes a container for AD accounts created for the machines that are to be created.

4. Set-BrokerCatalogMetadata -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -CatalogId 1 -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 -Name 'Citrix_DesktopStudio_IdentityPoolUid' -Value 'b99aee6d-8772-4dbc-978b-8eb9a26e2407'

Sets metadata on the Broker catalog with details of the Identity Pool. This is not essential.

5. Test-ProvSchemeNameAvailable -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -ProvisioningSchemeName @('ExampleMachines')

Checks that the requested name is available. This is not essential.

6. New-ProvScheme -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -CleanOnBoot -HostingUnitName 'SharedNFS' -IdentityPoolName 'ExampleMachines' -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 -MasterImageVM 'XDHyp:\hostingunits\SharedNFS\BaseVM.Base OS, domain joined and activated.snapshot\Pre-reqs installed.snapshot\Updates Applied.snapshot\VDA75-no agent.snapshot\Updated Agent.snapshot' -NetworkMapping @(0='xdhyp:\hostingunits\SharedNFS\Network 0.network') -PersonalVDiskDriveLetter P -PersonalVDiskDriveSize 10 -ProvisioningSchemeName 'ExampleMachines' -RunAsynchronously -Scope @() -UsePersonalVDiskStorage -VMCpuCount 1 -VMMemoryMB 1024

Creates a provisioning scheme object. This is a template for the machines that are to be created. It specifies the hypervisor, network, storage, memory, number of CPUs to be used etc. It takes parameters from the system already set up, such as the HostingUnit name and the path to the VM snapshot to be used for the machines to be created. This command makes a 'consolidated' copy of the VM snapshot being used and, as a result, the process can take time to complete.

In this example, the Studio script specified the -RunAsynchronous flag on this command. This means the command will return control to the administrator before it has completed, so you must wait for it to finish before performing any operations that require it to be complete. If this
flag is not specified, the command runs synchronously in-line and control is not returned until the command completes (successfully or otherwise). You can check the status of an asynchronous task using the Get-ProvTask cmdlet. Supply the task ID returned from the operation that started the task; in this case, the New-ProvScheme cmdlet.

7. Set-BrokerCatalog -AdminAddress 'ddc.dumdev.internal.citrix.com:80'
   -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 -Name 'ExampleMachines'
   -ProvisioningSchemeId 76125e3a-9001-4993-86b6-eefc85c87880

Updates the BrokerCatalog with the unique Id of the provisioning scheme created above.

8. Add-ProvSchemeControllerAddress -AdminAddress 'ddc.dumdev.internal.citrix.com:80'
   -ControllerAddress (@'DDC.dumdev.internal.citrix.com') -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46
   -ProvisioningSchemeName 'ExampleMachines'

Adds a set of controller addresses to the provisioning scheme object. This is a list of addresses that the machines created can use to register with a Controller (broker) when deployed. The machines’ registration addresses can be supplied in many ways; however, this information is required if the administrator wants to use the ‘Allow Machine Creation Service to supply this’ in the VDA installer. Changes to this list affect only machines created after the change, not existing machines.

   -IdentityPoolUid b99aee6d-8772-4dbc-978b-8eb9a26e2407 -Lock $False -MaxRecordCount 2147483647
   -State 'Available'

Studio gets a list of available Machine Identities from the Identity Pool so that, if existing accounts have been created in the past but are unused, these can be consumed instead of creating new accounts. Note that this is not required in a script because new accounts can be created instead, provided the script is running in a context that has permissions to do this. However, if the script does not have permissions to create accounts, change the script to consume available accounts (a separate process will be required to provide a pool of accounts into the Identity Pool, before running the script).

10. New-AcctADAccount -AdminAddress 'ddc.dumdev.internal.citrix.com:80'-Count 2
    -IdentityPoolUid b99aee6d-8772-4dbc-978b-8eb9a26e2407 -Lock $False -MaxRecordCount 2147483647
    -State 'Available'

Creates the required AD computer accounts in Active Directory. The script creates one account but, if required, it can create more using the 'Count' parameter of the command. The accounts are created into the OU defined in the provisioning scheme created above.

11. New-ProvVM -ADAccountName @('DUMDEV\Example-0001$','DUMDEV\Example-0002$')
    -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46
    -ProvisioningSchemeName 'ExampleMachines' -RunAsynchronously

Creates virtual machines, based on the template definition in the provisioning scheme created above. This process may take time to complete.

12. Lock-ProvVM -AdminAddress 'ddc.dumdev.internal.citrix.com:80'
    -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 -ProvisioningSchemeName 'ExampleMachines'
    -Tag 'Brokered' -VMID @('0710bb77-d01f-d006-4d67-5472e5cd349f')

Locks the provisioned virtual machines and prevents accidental modification of the virtual machine. Consumers of the SDK can use this to indicate that the virtual machine is in use and why it is locked. The script locks the VM with a tag of 'Brokered' to indicate the virtual machine is created and added to a Broker catalog and must not be deleted without first being removed from the catalog. You can set the Tag name to whatever is required.

13. New-BrokerMachine -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -CatalogUid 1
    -HostedMachineld '0710bb77-d01f-d006-4d67-5472e5cd349f' -HypervisorConnectionUid 1
    -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46
    -MachineName 'S-1-5-21-3918710733-2340574387-1999698698-109114'

Creates a Broker Machine object. These are objects stored in the catalog which join the provisioned machine with the catalog.

    -InputObject @(2) -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46

Requests the Broker Service to initiate a preparation operation for Personal vDisk. This is required to allow the machine to initialize the storage for Personal vDisk.

15. Stop-LogHighLevelOperation -AdminAddress 'ddc.dumdev.internal.citrix.com:80'
    -HighLevelOperationId f39a2792-064a-43eb-97c7-397cc1238e46 -IsSuccessful $true

Stops the logged operation begun in the first step and indicates it was successful.
Customize the script

The following section shows how to convert and adapt the Studio output into a script that is more consumable. In addition to using variables and removing commands that are not required, it shows how to add machine creation into a loop so that you can control the number of machines created. Line numbers have been added for readability.

1. `[CmdletBinding()]
   param
   (  
   [Parameter(Mandatory=$true)] [string] $hostingUnitPath,  
   [Parameter(Mandatory=$true)] [string] $catalogName,  
   [string] $catalogDescription,  
   [Parameter(Mandatory=$true)] [int] $numVmsToCreate,  
   [string] $adminAddress,  
   [Parameter(Mandatory=$true)] [string] $namingScheme,  
   [string] $OU,  
   [Parameter(Mandatory=$true)] [string] $domain,  
   [Parameter(Mandatory=$true)] [string] $masterImagePath  
   )
2. $hostingUnit = Get-Item $hostingUnitPath
3. $hostConnection = $hostingUnit.HypervisorConnection
4. $brokerHypConnection = Get-BrokerHypervisorConnection -HypervisorConnectionUid $hostConnection.HypervisorConnectionUid
5. # Start logged operation
6. $loggingOp = Start-LogHighLevelOperation -AdminAddress $adminAddress -Source 'Scripted' -Text "Create Machine Catalog '$catalogName'"
7. $loggingId = $loggingOp.Id
8. # Create the broker catalog and the AD Identity account pool
9. $catalog = New-BrokerCatalog -AllocationType 'Permanent' -Description $catalogDescription -IsRemotePC $False -MinimumFunctionalLevel 'L7' -Name $catalogName -PersistUserChanges 'OnPvd' -ProvisioningType 'MCS' -Scope @()  
   -SessionSupport 'SingleSession' -LoggingId $loggingId -AdminAddress $adminAddress
10. $adPool = New-AcctIdentityPool -IdentityPoolName $catalogName -NamingScheme $namingScheme -NamingSchemeType 'Numeric' -OU $OU -Domain $domain -AllowUnicode -LoggingId $loggingId -AdminAddress $adminAddress
11. Set-BrokerCatalogMetadata -CatalogId $catalog.Uid -Name 'Citrix_DesktopStudio_IdentityPoolUid' -Value $adPool.IdentityPoolUid -LoggingId $loggingId -AdminAddress $adminAddress
12. #create the ProvisioningScheme and wait for it to complete (reporting progress)
13. $provSchemeTaskID = New-ProvScheme -ProvisioningSchemeName $catalogName -HostingUnitUID $hostingUnit.HostingUnitUID -IdentityPoolUID $adPool.IdentityPoolUid -CleanOnBoot -MasterImageVM $masterImagePath -UsePersonalVDiskStorage -PersonalVDiskDriveLetter P -PersonalVDiskDriveSize 10 -RunAsynchronously -LoggingId $loggingId -AdminAddress $adminAddress
14. $ProvTask = get-provTask -TaskID $provSchemeTaskID -AdminAddress $adminAddress
15. $taskProgress = 0
16. write-host "Creating New ProvScheme"
17. while ($ProvTask.Active -eq $true)
18. {
19.  try
20.  {
21.   # catch an uninitialized task progress, this occurs until the product initialized the value
22.   $totalPercent = if ($provTask.TaskProgress){$provTask.TaskProgress} else {0}
23.  }
24.  catch
25.  {  
26.    Write-Progress -activity "Creating Provisioning Scheme:" -status "$totalPercent% Complete:" -percentcomplete $totalPercent
27.    sleep 30
28.  }
29.  $ProvTask = get-provTask -TaskID $provSchemeTaskID -AdminAddress $adminAddress
30. }
31. write-host "New ProvScheme Creation Finished"
28. $provScheme = get-provScheme -ProvisioningSchemeUID $provTask.ProvisioningSchemeUid
29. $controllers = Get-BrokerController | select DNSName
30. Add-ProvSchemeControllerAddress -ProvisioningSchemeUID $provScheme.ProvisioningSchemeUID -ControllerAddress $controllers -LoggingId $loggingId -AdminAddress $adminAddress

31. ###################################################################
32. # Set the provisioning scheme id for the broker catalog
33. Set-BrokerCatalog -InputObject $catalog -ProvisioningSchemeld $provTask.ProvisioningSchemeUid -LoggingId $loggingId -AdminAddress $adminAddress

34. ###################################################################
35. # create the AD accounts required and then create the Virtual machines (reporting progress)
36. $accts = New-AcctADAccount -IdentityPoolUid $adPool.IdentityPoolUid -Count $numVMsToCreate -LoggingId $loggingId -AdminAddress $adminAddress
37. $provVMTaskID = New-ProvVM -ProvisioningSchemeUID $provScheme.ProvisioningSchemeUID -ADAccountName $accts.SuccessfulAccounts -RunAsynchronously -LoggingId $loggingId -AdminAddress $adminAddress
38. # wait for the VMS to finish Provisioning
39. $ProvTask = get-provTask -TaskID $provVMTaskID -AdminAddress $adminAddress
40. while ($provTask.Active -eq $true)
41. {
42.    try {totalPercent = if ($provTask.TaskProgress){$provTask.TaskProgress} else {0}} catch {}
43.    Write-Progress -activity "Creating Machines:" -status "$totalPercent% Complete:" -percentcomplete $totalPercent
44.    sleep 5
45.    $ProvTask = get-provTask -TaskID $provVMTaskID -AdminAddress $adminAddress
46. }
47. write-host "VM Creation Finished"
48. # Lock the VMs and add them to the broker Catalog
49. $provisionedVMs = get-ProvVM -ProvisioningSchemeUID $provScheme.ProvisioningSchemeUID -AdminAddress $adminAddress
50. $provisionedVMs | Lock-ProvVM -ProvisioningSchemeUID $provScheme.ProvisioningSchemeUID -Tag 'Brokered' -LoggingId $loggingId -AdminAddress $adminAddress
51. $provisionedVMs | ForEach-Object {New-BrokerMachine -CatalogUid $catalog.UID -HostedMachineId $_.VMId -HypervisorConnectionUid $brokerHypConnection.UID -MachineName $_.ADAccountSid -LoggingId $loggingId -AdminAddress $adminAddress}
52. Stop-LogHighLevelOperation -IsSuccessful $true -HighLevelOperationId $loggingId -AdminAddress $adminAddress
Example: Create and configure a host

Oct 09, 2013

The following example shows how to create and configure a host.

Before you begin, make sure you follow the steps detailed in Get started with the XenDesktop SDK. This topic shows how to use Studio to perform the operation you want to script (in this case, to create a host) and collect the log of SDK operations that Studio made to perform the task. This output can then be customized to produce a script for automating host creation.

Note: To ensure you always get the latest enhancements and fixes, Citrix recommends you follow the procedure described in this topic, rather than copying and pasting the example script. Line numbers and line breaks have been added to the script for readability.

Understand the script

The following section explains what each part of the script produced by Studio is doing. This will help you with the customization of your own script. Line numbers have been added for readability.

1. ```
   Get-LogSite -AdminAddress 'mycontroller.example.com:80'
   ```

   Queries the configuration logging service to retrieve information about the site configuration.

2. ```
   Start-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80' -Source 'Studio'
   -StartTime 14/08/2013 14:30:28 -Text 'Create Connection `"Example XenServer"`'
   ```

   Starts a high-level logging operation with the configuration logging operation within which the rest of the commands will exist. Returns a log ID which is supplied to subsequent operations.

3. ```
   Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
   ```

   Sets the location of the Host Service that will be used by the configuration cmdlets. Because the Host Service exposes a PowerShell provider, not all of the cmdlets can take an address for the service so this cmdlet sets a default location.

4. ```
   New-Item  -ConnectionType 'XenServer' -HypervisorAddress @('http://xenhost1.example.com')
   -LoggingId e355ce51-8cbb-400a-ae81-1fdc567239cb -Path @('XDHyp:\Connections\Example XenServer')
   -Scope @() -Password ******** -UserName 'root'
   ```

   Creates a connection to a XenServer (xenhost1.example.com). This is a non-persistent connection and is available only to this PowerShell runspace.

5. ```
   Stop-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80' -EndTime 14/08/2013 14:30:29
   -HighLevelOperationId 'e355ce51-8cbb-400a-ae81-1fdc567239cb' -IsSuccessful $True
   ```

   Stops the logged operation begun previously and indicates it was successful.

6. ```
   Get-LogSite -AdminAddress 'mycontroller.example.com:80'
   ```

   Queries the configuration logging service to retrieve information about the site configuration.

7. ```
   Start-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80' -Source 'Studio'
   -StartTime 14/08/2013 14:30:30 -Text 'Update Connection `"Example XenServer"`'
   ```

   Starts a new high-level logging operation.

8. ```
   Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
   ```

   Sets the Host Service address details again (note that this repetition is removed in the optimized script below).

9. ```
   Set-Item -HypervisorAddress @('http://xenhost1.example.com','http://xenhost2.example.com')
   -LoggingId 44e15629-6906-4840-a36c-984aa607be6d -PassThru -Path @('XDHyp:\Connections\Example XenServer')
   -Password ******** -UserName 'root'
   ```

   Updates the connection created in step 4. Because there is more than one XenServer in the pool, it supplies all the addresses to enable High Availability.

10. ```
    Stop-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80' -EndTime 14/08/2013 14:30:31
    ```

    Stops the logged operation begun previously.
11. Get-LogSite -AdminAddress 'mycontroller.example.com:80'
Queries the configuration logging service to retrieve information about the site configuration.

12. Start-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80' -Source 'Studio'
-StartTime 14/08/2013 14:31:03 -Text 'Create Resources "Example Resources" and Persist Connection "Example XenServer"'
Starts a new logging operation.

13. Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
Sets the Host Service address details again.

14. Get-ChildItem -Path @('XDHyp:\Connections')
Gets the contents of the host connection to populate the wizard dialogs.

15. Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
Sets the Host Service address details again.

16. Remove-Item -LoggingId 76caa3f4-df93-4cb2-b78d-6a8824766314 -Path @('XDHyp:\Connections\Example XenServer')
Removes the temporary connection created in the wizard.

17. Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
Sets the Host Service address details again.

18. New-Item -ConnectionType 'XenServer' -HypervisorAddress @('http://xenhost1.example.com','http://xenhost2.example.com')
-LoggingId 76caa3f4-df93-4cb2-b78d-6a8824766314 -Path @('XDHyp:\Connections\Example XenServer') -Persist
-Scope @() -Password '********' -UserName 'root'
Recreates the connection as a persistent connection which is written to the database and available to other PowerShell runspaces.

19. New-BrokerHypervisorConnection -AdminAddress 'mycontroller.example.com:80'
-HypHypervisorConnectionUid a14096ba-5074-44ff-b596-371e345c0449 -LoggingId 76caa3f4-df93-4cb2-b78d-6a8824766314
Adds the host connection to the Broker Service.

20. Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
Sets the Host Service address details again.

21. New-Item -HypervisorConnectionName 'Example XenServer' -LoggingId 76caa3f4-df93-4cb2-b78d-6a8824766314
-NetworkPath @('XDHyp:\Connections\Example XenServer\Network 0.network') -Path @('XDHyp:\HostingUnits\Example Resources')
-PersonalvDiskStoragePath @('XDHyp:\Connections\Example XenServer\Pvd Storage.storage')
-RootPath 'XDHyp:\Connections\Example XenServer' -StoragePath @('XDHyp:\Connections\Example XenServer\Primary OS.storage')
Creates the HostingUnit (referred to as Resources in Studio) using the information gathered in step 14.

22. Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
Sets the Host Service address details again.

23. Get-Item -Path @('XDHyp:\Connections\Example XenServer')
Retrieves the newly created object.

Stop-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80' -EndTime 14/08/2013 14:31:07
-HighLevelOperationId '76caa3f4-df93-4cb2-b78d-6a8824766314' -IsSuccessful $True
Stops the logged operation begun previously and indicates if it was successful.

Customize the script

The following section shows how to convert and adapt the Studio output into a script that is more consumable. The following script has been simplified so that, instead of creating a temporary host connection in the process of acquiring information in the wizards as
in the Studio script above, a persistent connection is created. Information is then queried from within this to create the HostingUnit (Resources). Note that the LoggingId and HypHyperConnectionUid details are different.

Line numbers have been added for readability; each numbered item is a single PowerShell command.

1. Start-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80' -Source 'Studio' -Text 'Create Connection 'Example XenServer''
2. Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
3. New-Item -ConnectionType 'XenServer' -HypervisorAddress @('http://xenhost1.example.com', 'http://xenhost2.example.com') -LoggingId 76caa3f4-df93-4cb2-b78d-6a8824766314 -Path @('XDHyp:\Connections\Example XenServer') -Persist -Scope @() -Password ******** -UserName 'root'
4. Get-ChildItem -Path @('XDHyp:\Connections')
5. New-BrokerHypervisorConnection -AdminAddress 'mycontroller.example.com:80' -HypervisorConnectionUid a14096ba-5074-44ff-b596-371e345c0449 -LoggingId 76caa3f4-df93-4cb2-b78d-6a8824766314
6. New-Item -HypervisorConnectionName 'Example XenServer' -LoggingId 76caa3f4-df93-4cb2-b78d-6a8824766314 -NetworkPath @('XDHyp:\Connections\Example XenServer\Network 0.network') -Path @('XDHyp:\HostingUnits\Example Resources') -PersonalvDiskStoragePath @('XDHyp:\Connections\Example XenServer\Pvd Storage.storage') -RootPath 'XDHyp:\Connections\Example XenServer' -StoragePath @('XDHyp:\Connections\Example XenServer\PrimaryOS.storage')
7. Stop-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80' -HighLevelOperationId '76caa3f4-df93-4cb2-b78d-6a8824766314' -IsSuccessful $True
Example: Create a PvD Desktop

Sep 17, 2013

This topic provides an example of a script that creates a Delivery Group containing Personal vDisk (PvD) desktops.

Before you begin, make sure you follow the steps detailed in Get started with the XenDesktop SDK. This topic shows how to use Studio to perform the operation you want to script and collect the log of SDK operations that Studio made to perform the task. This output can then be customized to produce a script for automating the task.

**Note:** To ensure you always get the latest enhancements and fixes, Citrix recommends you follow the procedure described in this topic, rather than copying and pasting the example script.

**Understand the script**

The following section explains what each part of the script produced by Studio is doing. This will help you with the customization of your own script. Line numbers and line breaks have been added to the script for readability.


   Starts a logged operation and returns a log ID which is supplied to subsequent operations to associate them with the wider task.

2. New-BrokerDesktopGroup -AdminAddress 'test-ddc.mydomain.com:80' -ColorDepth 'TwentyFourBit' -DeliveryType 'DesktopsOnly' -DesktopKind 'Private' -InMaintenanceMode $False -IsRemotePC $False -LoggingId 8462d42-a994-4bce-ab58-be05c8d73b99 -MinimumFunctionalLevel 'L7' -Name 'Win7 PvD Desktops' -OffPeakBufferSizePercent 10 -PeakBufferSizePercent 10 -PublishedName 'Win7 PvD Desktops' -SecureIcaRequired $False -SessionSupport 'SingleSession' -ShutdownDesktopsAfterUse $False -TimeZone 'GMT Standard Time'

   Creates a new Delivery Group with options collected by the Studio wizard.

3. Add-BrokerMachinesToDesktopGroup -AdminAddress 'test-ddc.mydomain.com:80' -Catalog 'win7-pvd' -Count 2 -DesktopGroup 'Win7 PvD Desktops' -LoggingId 8462d42-a994-4bce-ab58-be05c8d73b99

   Adds the number of machines requested from the nominated catalog to the new Delivery Group.

4. Set-Variable -Name 'brokerUsers' -Value @('S-1-5-21-3291547628-200264090-930806513-1104','S-1-5-21-3291547628-200264090-930806513-1105')

   Get-BrokerUser -AdminAddress 'test-ddc.mydomain.com:80' -Filter {(SID -in $brokerUsers)} -MaxRecordCount 2147483647

   Remove-Variable -Name 'brokerUsers'

   New-BrokerUser -AdminAddress 'test-ddc.mydomain.com:80' -Name 'MYDOMAIN\user1'

   New-BrokerUser -AdminAddress 'test-ddc.mydomain.com:80' -Name 'MYDOMAIN\user2'

   The above commands are not required, Studio is verifying users.

5. Test-BrokerAssignmentPolicyRuleNameAvailable -AdminAddress 'test-ddc.mydomain.com:80' -Name @('Win7 PvD Desktops')

   Studio checks that the policy assignment name is available to use.


   Create the new policy assignment rule for the Delivery Group. No users are specified here so all control is through the access policy rule.

7. Set-Variable -Name 'brokerUsers' -Value @('S-1-5-21-3291547628-200264090-930806513-1104','S-1-5-21-3291547628-200264090-930806513-1105')

   Get-BrokerUser -AdminAddress 'test-ddc.mydomain.com:80' -Filter {(SID -in $brokerUsers)} -MaxRecordCount 2147483647

   Remove-Variable -Name 'brokerUsers'

   New-BrokerUser -AdminAddress 'test-ddc.mydomain.com:80' -Name 'MYDOMAIN\user1'

   New-BrokerUser -AdminAddress 'test-ddc.mydomain.com:80' -Name 'MYDOMAIN\user2'

   The above commands are not required, Studio is performing further checks.


   Studio tests that the access policy rule name is available to use.


   Log4j-enabled code generated to prevent spammy messages.
Creates the access policy rule for the new desktop for non-NetScaler Gateway connections.

           -AllowRestart $True -DesktopGroupUid 41 -Enabled $True -IncludedSmartAccessFilterEnabled $True -IncludedSmartAccessTags @()
           -IncludedUserFilterEnabled $True -IncludedUsers @('MYDOMAIN\user1','MYDOMAIN\user2')
           -LoggingId 846f2d42-a994-4be6-be05c8d73b99 -Name 'Win7 PvD Desktops_AG'
Studio repeats this process for NetScaler Gateway connections.

11. Test-BrokerPowerTimeSchemeNameAvailable  -AdminAddress 'test-ddc.mydomain.com:80' -Name @('Win7 PvD Desktops_Weekdays')
    New-BrokerPowerTimeScheme  -AdminAddress 'test-ddc.mydomain.com:80' -DaysOfWeek 'Weekdays' -DesktopGroupUid 41 -DisplayName 'Weekdays'
           -LoggingId 846f2d42-a994-4be6-be05c8d73b99 -Name 'Win7 PvD Desktops_Weekdays'
           -PeakHours @($False,$False,$False,$False,$False,$False,$False,$True,$True,$True,$True,$True,$True,$True,$True,$True,$True,$True,$True,$True,$False,$False,$False,$False,$False)
           -PoolSize @(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
    Test-BrokerPowerTimeSchemeNameAvailable  -AdminAddress 'test-ddc.mydomain.com:80' -Name @('Win7 PvD Desktops_Weekend')
    New-BrokerPowerTimeScheme  -AdminAddress 'test-ddc.mydomain.com:80' -DaysOfWeek 'Weekend' -DesktopGroupUid 41
           -Displayname 'Weekend' -LoggingId 846f2d42-a994-4be6-be05c8d73b99 -Name 'Win7 PvD Desktops_Weekend'
           -PeakHours @($False,$False,$False,$False,$False,$False,$False,$True,$True,$True,$True,$True,$True,$True,$True,$True,$True,$True,$True,$True,$False,$False,$False,$False,$False)
           -PoolSize @(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
Studio checks that the names for the (optional) weekday and weekend power schemes are available, and adds these.

    -HighLevelOperationId '846f2d42-a994-4be6-be05c8d73b99' -Name '@(Win7 PvD Desktops)' -IsSuccessful $True
Stops the logged operation begun in step 1 and indicates it was successful.

Customize the script

This section shows how to convert and adapt the Studio output into a script that is more consumable.

The script creates a Delivery Group containing PvD desktops. The catalog specified in the parameters must exist already and be populated appropriately (with an allocation type of static and a PvD disk). The script is designed to be run from a PowerShell command line logged on as a Citrix administrator. No checks are made for permissions; the script will fail if the user does not have the appropriate permissions.

<#
    Sample usage:
    .\CreatePvDGroup.ps1
    -GroupName "Win7 PvD Desktops"
    -SrcCatalog "win7-pvd"
    -NumDesktops 2
    -Users @('mydomain\user1','mydomain\user2')
#>
Param(
    [Parameter(Mandatory=$true)] [string] $GroupName,
    [Parameter(Mandatory=$true)] [string] $SrcCatalog,
    [Parameter(Mandatory=$true)] [int] $NumDesktops,
    [Parameter(Mandatory=$true)] [array] $Users
)

The table explains the parameters used in the script.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GroupName</td>
<td>The name of the new PvD desktop group</td>
</tr>
<tr>
<td>SrcCatalog</td>
<td>The name of the catalog to be used to create the PvD desktop. Create the catalog by specifying an allocation type of static. Machines must also have PvD disks.</td>
</tr>
<tr>
<td>NumDesktops</td>
<td>The number of machines to add to the PvD desktop group. If insufficient machines are available, as many as possible are added.</td>
</tr>
<tr>
<td>Users</td>
<td>Which users can access the group. This is a list of users or groups; for example, @{mydomain\Domain Users} or @{mydomain\user1;mydomain\user2}</td>
</tr>
</tbody>
</table>
Set-HypAdminConnection -AdminAddress $adminAddress

Specify the hypervisor admin connection to use. Removes the need for the -AdminAddress for some of the commands.

$peakPoolSize = 2
$weekendPoolSizeByHour = new-object int[24]
$weekdayPoolSizeByHour = new-object int[24]
9..17 | %{ $weekdayPoolSizeByHour[$_] = $peakPoolSize }
$peakHours = (0..23 | %{ $_ -ge 9 -and $_ -le 17 })

This creates 24 element arrays with a 1 or a 0 in each entry. Use these to specify when peak hours are for the power schedules for the Delivery Groups. Elements 9 to 17 (hours starting 09:00 to 17:00) for weekdays are set to 1, others are left at 0. Two unassigned machines are powered up during peak times, if available.

$logId = Start-LogHighLevelOperation
  -Text "Create PvD desktop group"
  -Source "Create PvD Desktop Group Script"

Start a new logged operation. This returns a log ID which is passed into subsequent operations to associate them with the create group task.

$grp = New-BrokerDesktopGroup
  -DesktopKind 'Private'
  -DeliveryType 'DesktopsOnly'
  -LoggingId $logId.Id
  -Name $GroupName
  -PublishedName $GroupName
  -SessionSupport 'SingleSession'
  -ShutdownDesktopsAfterUse $False

$count = Add-BrokerMachinesToDesktopGroup
  -Catalog $SrcCatalog
  -Count $NumDesktops
  -DesktopGroup $GroupName
  -LoggingId $logId.Id

"$count machines added to the PvD desktop group"

Create the new Delivery Group, delivering private desktops. The catalog used must have been populated with suitable machines (permanent with a PvD disk). PublishedName is the name seen by end users; the following uses the same name as the group name.

New-BrokerAssignmentPolicyRule
  -DesktopGroupUid $grp.Uid
  -IncludedUserFilterEnabled $False
  -LoggingId $logId.Id
  -MaxDesktops 1
  -Name ($GroupName + '_AssignRule')
  | Out-Null

Assigned desktops need an assignment policy. Disable user filter so that access is controlled entirely by access policy rules.

New-BrokerAccessPolicyRule
  -AllowedConnections 'NotViaAG'
  -AllowedProtocols @('HDX','RDP')
  -AllowRestart $True
  -DesktopGroupUid $grp.Uid
  -IncludedSmartAccessFilterEnabled $True
  -IncludedUserFilterEnabled $True
  -IncludedUsers $Users
  -LoggingId $logId.Id
  -Name ($GroupName + '_Direct')
  | Out-Null

New-BrokerAccessPolicyRule
  -AllowedConnections 'ViaAG'
  -AllowedProtocols @('HDX','RDP')
  -AllowRestart $True
  -DesktopGroupUid $grp.Uid
  -IncludedSmartAccessFilterEnabled $True
  -IncludedUserFilterEnabled $True
  -IncludedSmartAccessTags @()
Specify any access restrictions: allow direct access using NetScaler Gateway, using HDX & RDP protocols. The user can request the desktop be restarted, if necessary.

```
New-BrokerPowerTimeScheme
    -DaysOfWeek 'Weekdays'
    -DesktopGroupUid $grp.Uid
    -DisplayName 'Weekdays'
    -LoggingId $logId.Id
    -Name ($GroupName + '_Weekdays')
    -PeakHours $peakHours
    -PoolSize $weekdayPoolSizeByHour
    | Out-Null
New-BrokerPowerTimeScheme
    -DaysOfWeek 'Weekend'
    -DesktopGroupUid $grp.Uid
    -DisplayName 'Weekend'
    -LoggingId $logId.Id
    -Name ($GroupName + '_Weekend')
    -PeakHours $peakHours
    -PoolSize $weekendPoolSizeByHour
    | Out-Null
```

Optional: Specify power schedules.

```
Stop-LogHighLevelOperation -HighLevelOperationId $logId.Id -IsSuccessful $True
Stop configuration logging and indicate if successful or not.
```
Example: Get load balancing information

Mar 28, 2015
In XenDesktop 7, you can use Server OS Machines to deliver cost-effective applications and desktops hosted on server operating systems to multiple users.

To load balance Server OS Machines in a XenDesktop 7 deployment, you use Citrix Policies. There are several load balancing policies for enabling and configuring load management between servers delivering Windows Server OS machines. For more information about these policies, see Load Management policy settings. You work with policies through Studio or the Group Policy Management Console in Windows. For more information about working with policies, see Manage Citrix policies.

To see the load, you can use either the Citrix Director or Studio consoles, or the PowerShell SDK. The following example shows how to use the PowerShell SDK to display the load.

Note: If you've used previous versions of XenDesktop, you may be familiar with the qfarm/load command. This tool is no longer available in XenDesktop 7, but you can use PowerShell to display similar output as shown in the example below.

Example: Get load index values

To display a list of machines with their calculated/measured load index values, together with counts of sessions running on them:
1. Start a shell in PowerShell. For more information, see About the XenDesktop SDK.
2. Type:
   ```powershell
   Get-BrokerMachine -SessionSupport MultiSession -Property 'DnsName','LoadIndex','SessionCount'
   ```

   For more information and examples, see the cmdlet help for the get-brokermachine cmdlet and About topics such as about_broker_filtering-xd7.html. See XenDesktop cmdlet help.
XenDesktop cmdlet help

Oct 02, 2013
This section provides the PowerShell help text for all the XenDesktop 7 cmdlets.

- Citrix.AdIdentity.Admin.V2
- Citrix.AppV.Admin.V1
- Citrix.Broker.Admin.V2
- Citrix.Configuration.Admin.V2
- Citrix.ConfigurationLogging.Admin.V1
- Citrix.DelegatedAdmin.Admin.V1
- Citrix.EnvTest.Admin.V1
- Citrix.Host.Admin.V2
- Citrix.Monitor.Admin.V1
- Citrix.Storefront.Admin.V1
Citrix.AdIdentity.Admin.V2

Oct 10, 2013

Overview

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcctAdIdentitySnapin</td>
<td>The Active Directory Identity Service PowerShell snap-in provides</td>
</tr>
<tr>
<td>Acct Filtering</td>
<td>Describes the common filtering options for XenDesktop cmdlets.</td>
</tr>
</tbody>
</table>

Cmdlets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-AcctADAccount</td>
<td>Import Active Directory computer accounts from Active Directory for use in the AD Identity Service.</td>
</tr>
<tr>
<td>Add-AcctIdentityPoolScope</td>
<td>Add the specified IdentityPool(s) to the given scope(s).</td>
</tr>
<tr>
<td>Copy-AcctIdentityPool</td>
<td>Copies an Identity Pool and its associated Identities to a new IdentityPool</td>
</tr>
<tr>
<td>Get-AcctADAccount</td>
<td>Gets the AD accounts stored in the AD Identity Service.</td>
</tr>
<tr>
<td>Get-AcctDBCConnection</td>
<td>Gets the database string for the specified data store used by the AdIdentity Service.</td>
</tr>
<tr>
<td>Get-AcctDBSchema</td>
<td>Gets a script that creates the AdIdentity Service database schema for the specified data store.</td>
</tr>
<tr>
<td>Get-AcctDBVersionChangeScript</td>
<td>Gets a script that updates the AdIdentity Service database schema.</td>
</tr>
<tr>
<td>Get-AcctIdentityPool</td>
<td>Gets existing identity pools.</td>
</tr>
<tr>
<td>Get-AcctInstalledDBVersion</td>
<td>Gets a list of all available database schema versions for the AdIdentity Service.</td>
</tr>
<tr>
<td>Get-AcctScopedObject</td>
<td>Gets the details of the scoped objects for the AdIdentity Service.</td>
</tr>
<tr>
<td>Get-AcctService</td>
<td>Gets the service record entries for the AdIdentity Service.</td>
</tr>
<tr>
<td>Get-AcctServiceAddedCapability</td>
<td>Gets any added capabilities for the AdIdentity Service on the controller.</td>
</tr>
<tr>
<td>Get-AcctServiceInstance</td>
<td>Gets the service instance entries for the AdIdentity Service.</td>
</tr>
<tr>
<td>Get-AcctServiceStatus</td>
<td>Gets the current status of the AdIdentity Service on the controller.</td>
</tr>
<tr>
<td>New-AcctADAccount</td>
<td>Creates AD computer accounts in the specified identity pool.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------</td>
</tr>
<tr>
<td>New-AcctIdentityPool</td>
<td>Creates a new identity pool.</td>
</tr>
<tr>
<td>Remove-AcctADAccount</td>
<td>Removes AD computer accounts from an identity pool.</td>
</tr>
<tr>
<td>Remove-AcctIdentityPool</td>
<td>Removes identity pools.</td>
</tr>
<tr>
<td>Remove-AcctIdentityPoolMetadata</td>
<td>Removes metadata from the given IdentityPool.</td>
</tr>
<tr>
<td>Remove-AcctIdentityPoolScope</td>
<td>Remove the specified IdentityPool(s) from the given scope(s).</td>
</tr>
<tr>
<td>Remove-AcctServiceMetadata</td>
<td>Removes metadata from the given Service.</td>
</tr>
<tr>
<td>Rename-AcctIdentityPool</td>
<td>Renames an identity pool.</td>
</tr>
<tr>
<td>Repair-AcctADAccount</td>
<td>Resets the AD machine password for the given accounts.</td>
</tr>
<tr>
<td>Reset-AcctServiceGroupMembership</td>
<td>Reloads the access permissions and configuration service locations for the AdIdentity Service.</td>
</tr>
<tr>
<td>Set-AcctDBConnection</td>
<td>Configures a database connection for the AdIdentity Service.</td>
</tr>
<tr>
<td>Set-AcctIdentityPool</td>
<td>Update parameters of an identity pool.</td>
</tr>
<tr>
<td>Set-AcctIdentityPoolMetadata</td>
<td>Adds or updates metadata on the given IdentityPool.</td>
</tr>
<tr>
<td>Set-AcctServiceMetadata</td>
<td>Adds or updates metadata on the given Service.</td>
</tr>
<tr>
<td>Test-AcctDBConnection</td>
<td>Tests a database connection for the AdIdentity Service.</td>
</tr>
<tr>
<td>Test-AcctIdentityPoolNameAvailable</td>
<td>Checks to ensure that the proposed name for an identity pool is unused.</td>
</tr>
<tr>
<td>Unlock-AcctADAccount</td>
<td>Unlocks AD accounts within the AD Identity Service.</td>
</tr>
<tr>
<td>Unlock-AcctIdentityPool</td>
<td>Unlocks identity pools.</td>
</tr>
<tr>
<td>Update-AcctADAccount</td>
<td>Refreshes the AD computer account state stored in the AD Identity Service.</td>
</tr>
</tbody>
</table>
about_AcctAdIdentitySnapin

Oct 10, 2013

TOPIC

about_AcctADIdentityServiceSnapin

SHORT DESCRIPTION

The Active Directory Identity Service PowerShell snap-in provides administrative functions for the Active Directory Identity Service.

COMMAND PREFIX

All commands in this snap-in have 'Acct' in their name.

LONG DESCRIPTION

The Active Directory Identity Service PowerShell snap-in enables both local and remote administration of the Active Directory Identity Service. It provides facilities to store details about Active Directory computer accounts that the Machine Creation Service can use.

The snap-in provides two main entities:

Identity

A representation of an Active Directory computer account that reflects the state of the account within the context of the Machine Creation Service. When an account is created by or imported into the Active Directory Identity Service, the account password is stored. Once the account is consumed by the Machine Creation Service, the password is discarded. For accounts registered with the Active Directory Identity Service, identities hold the following additional state information.

Available
The Active Directory account is registered with the service, the password for the account is known, and the account is available to be consumed by another service. Accounts that are successfully created with the New-AcctADAccount command or imported using the Add-ADAccount command, are initially assigned this state.

InUse
The Active Directory account is registered and has been consumed by another service. The password for the account is no longer known to the service.

Error
The Active Directory account is registered, but is missing.
disabled, or locked within Active Directory. Accounts that are not successfully created with the New-AcctADAccount command or imported using the Add-ADAccount command appear in this state. Use the Update-AcctADAccount and Repair-AcctADAccount commands to resolve issues with accounts in this state.

Tainted
The Active Directory account is registered and has been released by all the consuming services, but cannot be made available for use as the password is no longer known. Use the Repair-AcctADAccount command to reset account passwords and restore the account state to 'Available'.

Identities can also be marked as 'Locked' by the Machine Creation Service to indicate that they are in use and must not be changed. These services are also responsible for unlocking the Active Directory accounts when they no longer require exclusive access. Use the Unlock-AcctADAccount command to allow the lock to be overridden, if necessary.

Identity Pool

Containers for identities that can be configured with all the information required for new Active Directory accounts to be created. Alternatively, identity pools can be populated by importing accounts that already exist in Active Directory. All accounts registered with the Active Directory Identity Service must be placed into one of these containers. An identity can belong to more than one identity pool, but the state of the identity cannot be different in each pool. For example, an identity that is in use will be marked 'InUse' in all the identity pools of which it is part.

To avoid conflicting changes, identity pools can also be marked as 'Locked' during operations that modify the content of a pool. These operations are also responsible for unlocking the identity pool. Use the unlock-AcctIdentityPool command to allow the lock to be overridden, if necessary.

ACTIVE DIRECTORY PERMISSIONS

Account Creation (using the New-AcctADAccount command)

To use PowerShell to create new Active Directory accounts, the runspace
must be run using an account with sufficient permissions in the required Active Directory container (specified by the identity pool organizational unit parameter) for accounts to be created.

Import Accounts (using the Add-AcctADAccount command)

There are two modes for this operation: situations where the Active Directory account passwords are known and situations where the passwords are not known.

If the account passwords are known, the accounts can be imported without the need for administrative permissions in Active Directory. The accounts are imported and the password provided is used to change the existing password.

If the passwords are not known, the runspace must be run using an account that has permissions to reset the password for the accounts.
about_Acct_Filtering

Oct 10, 2013

TOPIC

XenDesktop - Advanced Dataset Filtering

SHORT DESCRIPTION

Describes the common filtering options for XenDesktop cmdlets.

LONG DESCRIPTION

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

```
Get-<Noun> | Where { $_.Size = 'Small' } | Sort 'Date' | Select -First 10
```

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

```
Get-<Noun> -Size 'Small' -SortBy 'Date' -MaxRecordCount 10
```

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

```
-MaxRecordCount <int>
```

Specifies the maximum number of results to return.

For example, to return only the first nine results use:

```
Get-<Noun> -MaxRecordCount 9
```

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:
WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

To retrieve all records, specify a large number for -MaxRecordCount. As the value is an integer, you can use the following:

```
Get-<Noun> -MaxRecordCount [int]:MaxValue
```

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
```

```
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
  + Get-<Noun> <<<< -MaxRecordCount 9 -ReturnTotalRecordCount
  + CategoryInfo : OperationStopped: () [Get-<Noun>], PartialDataException
  + FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

```
$count = $error[0].TotalAvailableResultCount
```

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semicolons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.
Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-sortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values. Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

-sortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'

-filter <string>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter 'Name -like "High*" -or (Priority -eq 1 -and Severity -ge 2)'

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company "citrix" -Product '[X]EN''
Get-<Noun> -Product "Xen**" -Company "CITRIX"
Get-<Noun> -Filter { Company -eq 'Citrix' -and Product -like 'Xen'* }
See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```powershell
Get-<Noun> -Company "Abc[*]"                 # Matches Abc*
Get-<Noun> -Company "Abc"**                  # Matches Abc*
Get-<Noun> -Filter { Company -eq "Abc"** }  # Matches Abc*
Get-<Noun> -Filter { Company -eq "A""B""C" } # Matches A"B"C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```powershell
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

```powershell
Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like '[123]''
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```powershell
Get-<Noun> -Filter 'Enabled'      # Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled' # Equivalent to 'Enabled -eq $false'
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```powershell
Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:
```powershell
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle" }
Get-<Noun> -Filter { Shape -like 'C' }
```

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

```
Get-<Noun> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z" }
$d = [DateTime]"2010-08-23T12:30:00.0Z"
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }
```

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

```
Get-<Noun> -Filter { StartTime -ge "-2" }   # Two days ago
Get-<Noun> -Filter { StartTime -ge "-1:30" } # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge "-0:0:30" } # 30 seconds ago
```

**ARRAY PROPERTIES**

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the `-contains` and `-notcontains` operators. Unlike PowerShell, these perform wildcard matching on strings.

Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

```
Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*" }
Get-<Noun> -Filter { Users -contains "Fred*" }
```

You can also use the singular form with `-Filter` to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter { User -eq "Frederick" }
# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter { User -lt 'F' }
```

**COMPLEX EXPRESSIONS**
When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

```powershell
Get-<Noun> -Filter { Shape -eq 'Circle' -or Shape -eq 'Square' }
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }
```

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

```powershell
Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue' -and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue' -and Shape -eq 'Circle') }
```

**PAGING**

The simplest way to page through data is to use the -Skip and -MaxRecordCount parameters. So, to read the first three pages of data with 10 records per page, use:

```powershell
Get-<Noun> -Skip 0 -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 10 -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 20 -MaxRecordCount 10 <other filtering criteria>
```

You must include the same filtering criteria on each call, and ensure that the data is sorted consistently.

The above approach is often acceptable, but as each call performs an independent query, data changes can result in records being skipped or appearing twice. One approach to improve this is to sort by a unique id field and then start the search for the next page at the unique id after the last unique id of the previous page. For example:

```powershell
# Get the first page
Get-<Noun> -MaxRecordCount 10 -SortBy SerialNumber

SerialNumber ... 
---------- ...
A120004
A120007
... 7 other records ...
A120900

# Get the next page
Get-<Noun> -MaxRecordCount 10 -Filter { FirstName -gt 'A120900' }

SerialNumber ...
```
FILTER SYNTAX DEFINITION

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= "{" <ComponentList> "}"

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

   <Component>

<Component> ::= <NotOperator> <Factor> |

   <Factor>

<Factor> ::= "(" <ComponentList> ")" |

   <PropertyName> <ComparisonOperator> <Value> |
   <PropertyName>

<AndOrOperator> ::= "-and" | "-or"

<NotOperator> ::= "-not" | "!"

<ComparisonOperator>

   ::= "-eq" | "-ne" | "-le" | "-ge" | "-lt" | "-gt" |
   "-like" | "-notlike" | "-contains" | "-notcontains" |
   "-in" | "-notin"

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

   <scalar variable> | <array variable> |
   "$null" | "$true" | "$false"
Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30" means 1 hour and 30 minutes ago.
Add-AcctADAccount

Oct 10, 2013
Import Active Directory computer accounts from Active Directory for use in the AD Identity Service.

Syntax


Add-AcctADAccount -IdentityPoolUid <Guid> -ADAccountName <String[]> [-Password <String>] [-SecurePassword <SecureString>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability for AD computer accounts that already exist in AD to be used by the Citrix AD Identity Service and the other Citrix Machine Creation Services.

All aspects of this command that need to make modifications to the accounts in AD will do so using the account that the run space is using. This means that if the passwords need resetting for the accounts, the user performing the operation in PowerShell must have sufficient privileges in AD for this operation to complete successfully.

The following rules apply to the importing of AD accounts; If a password is supplied, the account will be imported. The cmdlet will attempt to change the password so that it is known only to the Citrix Identity Service. This uses password change operations and does not need AD account administration permissions. If a password is not supplied, the cmdlet will attempt to reset the password for the AD account. This requires the cmdlet to have enough privileges in AD for the accounts' password reset to be available. Imported accounts in a disabled or locked state in AD are imported with the account marked in an error state. If the identity pool into which the account is being imported does not have a domain set, it assumes the domain of the first account imported into it.

Related topics

New-AcctADAccount
Remove-AcctADAccount
Repair-AcctADAccount
Get-AcctADAccount
Update-AcctADAccount
Unlock-AcctADAccount

Parameters

-IdentityPoolName <String>

The identity pool name into which to add the imported accounts.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
**-ADAccountName**: <String[]>

The AD account name to be imported.

AD accounts are accepted in the following formats: Fully qualified DN e.g. CN=MyComputer,OU=Computers,DC=MyDomain,DC=Com; UPN format e.g. MyComputer@MyDomain.Com; Domain qualified e.g. MyDomain\MyComputer.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | false |

**-IdentityPoolUid**: <Guid>

The unique identifier for the identity pool to which imported accounts are to be added.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByPropertyName) |

**-Password**: <String>

The password for the computer account.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-SecurePassword**: <SecureString>

The password for the account (provided in a Secure String class).

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-LoggingId**: <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
**Default Value**

| Accept Pipeline Input? | false |

---

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |

| Default Value | LocalHost. Once a value is provided by any cmdlet, this value becomes the default. |

| Accept Pipeline Input? | false |

---

**Return Values**

Citrix.AdIdentity.Sdk.AccountOperationDetailedSummary

The Add-AcctADAccount returns an object that contains the following parameters:

- **SuccessfulAccountsCount**<int>
  The number of accounts that were added successfully.

- **FailedAccountsCount**<int>
  The number of accounts that were not added.

- **FailedAccounts**<Citrix.XDPowerShell.AccountError[]>
  The list of accounts that failed to be added. Each one has the following properties:
  - **ADAccountName**<string>
  - **ADAccountSid**<String>
  - **ErrorReason**<AdIdentityStatus>
    This can be one of the following:
    - **UnableToConvertDomain**
    - **UnableToAccessAccountProperties**
    - **IdentityNotLocatedInDomain**
    - **UnableToAccessAccountProperties**
    - **IdentityDuplicateObjectExists**
    - **IdentityObjectLocked**
    - **IdentityObjectInUse**
    - **FailedToConnectToDomainController**
FailedToExecuteSearchInAD
FailedToAccessComputerAccountInAD
FailedToSetPasswordInAD
FailedToChangePasswordInAD
ADServiceDatabaseError
ADServiceDatabaseNotConfigured
ADServiceStatusInvalidDb
DiagnosticInformation <Exception>
Any other error information
SuccessfulAccounts <Citrix.AdIdentity.Sdk.Identity[]>
The list of accounts that were successfully added. Each one has the following properties:
ADAccountSid <string>
The AD account SID for the imported account.
ADAccountName <string>
The AD account name for the imported account.
Domain <string>
The domain for the imported account.
State <Citrix.AdIdentity.Sdk.ADIentityState>
The state for the account. This can be:
Available
The account is not used.
InUse
The account is in use.
Error
The account is in error (i.e. the account is locked or disabled in AD).
Tainted
The account is no longer used, but the password is no longer known.
Lock <Boolean>
The account is locked (in the database not in AD).
Notes
To maintain maximum security when using the command programmatically, Citrix recommends you use the 'SecurePassword' property instead of the 'Password' property.

In the case of failure, the following errors can result.

Error Codes
-----------
IdentityPoolAlreadyLocked
The specified identity pool was locked by another operation.

IdentityPoolNotFound
The specified identity pool was not found.

IdentityDuplicateObjectExists
The specified AD account already exists for the identity pool.

PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
C:\PS>Add-AcctADAccount -IdentityPoolName MyPool -ADAccountName "Domain\account","Domain\account2" -OutVariable result

SuccessfulAccounts  SuccessfulAccountsCount  FailedAccountsCount  FailedAccounts
| Domain\account, domain\account2 | 2 | 0 | {} |

```powershell
$result[0].SuccessfulAccounts

ADAccountSid : S-1-5-21-1315084875-1285793635-2418178940-2644
ADAccountName : domain\account
Domain        : Domain.com
State         : Available
Lock          : False

ADAccountSid : S-1-5-21-1315084875-1285793635-2418178940-2645
ADAccountName : domain\account2
Domain        : Domain.com
State         : Available
Lock          : False
```

Import the two accounts (account and account2) from AD into the identity Pool called "MyPool"
Add-AcctIdentityPoolScope

Add the specified IdentityPool(s) to the given scope(s).

Syntax

Add-AcctIdentityPoolScope [-Scope] <String[]> -InputObject <IdentityPool[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Add-AcctIdentityPoolScope [-Scope] <String[]> -IdentityPoolUid <Guid[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Add-AcctIdentityPoolScope [-Scope] <String[]> -IdentityPoolName <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The AddAcctIdentityPoolScope cmdlet is used to associate one or more IdentityPool objects with given scope(s).

There are multiple parameter sets for this cmdlet, allowing you to identify the IdentityPool objects in different ways:

- IdentityPool objects can be piped in or specified by the InputObject parameter
- The IdentityPoolUid parameter specifies objects by IdentityPoolUid
- The IdentityPoolName parameter specifies objects by IdentityPoolName (supports wildcards)

To add a IdentityPool to a scope you need permission to change the scopes of the IdentityPool and permission to add objects to all of the scopes you have specified.

If the IdentityPool is already in a scope, that scope will be silently ignored.

Related topics

Remove-AcctIdentityPoolScope

Get-AcctScopedObject

Parameters

-Scope<String[]>

Specifies the scopes to add the objects to.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-InputObject<IdentityPool[]>
Specifies the IdentityPool objects to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

**-IdentityPoolUid**<Guid[]>

Specifies the IdentityPool objects to be added by IdentityPoolUid.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

**-IdentityPoolName**<String[]>

Specifies the IdentityPool objects to be added by IdentityPoolName.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values
None

Notes
If the command fails, the following errors can be returned.

Error Codes
--------

UnknownObject
One of the specified objects was not found.

ScopeNotFound
One of the specified scopes was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS> Add-AcctIdentityPoolScope Finance -IdentityPoolUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
Adds a single IdentityPool to the 'Finance' scope.

-------------------------- EXAMPLE 2 --------------------------
c:\PS> Add-AcctIdentityPoolScope Finance,Marketing -IdentityPoolUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
Adds a single IdentityPool to the multiple scopes.

-------------------------- EXAMPLE 3 --------------------------
c:\PS> Get-AcctIdentityPool | Add-AcctIdentityPoolScope Finance
Adds all visible IdentityPool objects to the 'Finance' scope.

-------------------------- EXAMPLE 4 --------------------------
c:\PS> Add-AcctIdentityPoolScope Finance -IdentityPoolName A*
Adds IdentityPool objects with a name starting with an 'A' to the 'Finance' scope.
Copy-AcctIdentityPool

Oct 10, 2013
Copies an Identity Pool and its associated Identities to a new IdentityPool

Syntax

Copy-AcctIdentityPool [-IdentityPoolName] <String> [-NewIdentityPoolName] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Copy-AcctIdentityPool -IdentityPoolUid <Guid> [-NewIdentityPoolName] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to copy an IdentityPool.

The new IdentityPool will contain all the accounts that were in the original pool and will have the same domain and OU set. The naming scheme will be unset and the StartCount will be set to 1.

Related topics

New-AcctIdentityPool
Get-AcctIdentityPool
Remove-AcctIdentityPool

Parameters

- **IdentityPoolName**<String>
  The name of the identity pool that is to be copied.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **IdentityPoolUid**<Guid>
  The unique identifier for the identity pool that is to be copied.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **NewIdentityPoolName**<String>
The name for the new IdentityPool.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Return Values

Citrix.AdIdentity.Sdk.IdentityPool

This object provides details of the new identity pool and contains the following information:

IdentityPoolName <string>

The name of the identity pool.

IdentityPoolUid <Guid>

The unique identifier for the identity pool.

NamingScheme <string>

The naming scheme for the identity pool.

NamingSchemeType <Citrix.XDInterServiceTypes.ADIdentityNamingScheme>

The naming scheme type for the identity pool. This can be one of the following:
Numeric - naming scheme uses numeric indexes

Alphabetic - naming scheme uses alphabetic indexes

StartCount <int>
The next index to be used when creating an identity from the identity pool.

OU <string>
The Active Directory distinguished name for the OU in which accounts created from this identity pool will be created.

Domain <string>
The Active Directory domain that accounts in the pool belong to.

Lock <Boolean>
Indicates whether the identity pool is locked.

Notes
In the case of failure, the following errors can result.

Error Codes
----------

IdentityPoolDuplicateObjectExists
An identity pool with the same name exists already.

IdentityPoolObjectNotFound
The identity pool to be modified could not be located.

PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for
for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.
Get-AcctADAccount

Oct 10, 2013
Gets the AD accounts stored in the AD Identity Service.

Syntax

```
```

```
```

Detailed Description

Provides the ability to locate the AD accounts stored within the AD Identity Service and view the state of the accounts.

Related topics

- New-AcctADAccount
- Add-AcctADAccount
- Remove-AcctADAccount
- Unlock-AcctADAccount
- Update-AcctADAccount
- Repair-AcctADAccount

Parameters

**-ADAccountSid<String>**
The AD Account SID of the account.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

**-Domain<String>**
The domain of the account (this is in dns format).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>State</strong>&lt;ADIdentityState&gt;</td>
<td>The current state of the identity stored in the AD Identity Service for the AD account.</td>
</tr>
<tr>
<td><strong>Lock</strong>&lt;Boolean&gt;</td>
<td>Indicates if the account is locked in the AD Identity Service.</td>
</tr>
<tr>
<td><strong>ReturnTotalRecordCount</strong>&lt;SwitchParameter&gt;</td>
<td>See about_Acct_Filtering for details.</td>
</tr>
<tr>
<td><strong>MaxRecordCount</strong>&lt;Int32&gt;</td>
<td>See about_Acct_Filtering for details.</td>
</tr>
<tr>
<td><strong>Skip</strong>&lt;Int32&gt;</td>
<td>See about_Acct_Filtering for details.</td>
</tr>
</tbody>
</table>
**Accept Pipeline Input?**

- **SortBy**: `<String>`
  
  See about _Acct Filtering_ for details.

- **Filter**: `<String>`
  
  See about _Acct Filtering_ for details.

- **AdminAddress**: `<String>`

  Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

- **IdentityPoolName**: `<String>`

  The name of the identity pool to which the account is registered.

- **IdentityPoolUid**: `<Guid>`

  The unique identifier for the identity pool that the account is registered to.
<table>
<thead>
<tr>
<th>Default Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**

`Citrix.AdIdentity.Sdk.IdentityInPool`

The `Get-AcctADAccount` returns an object that contains the following parameters:

- **ADAccountSid <string>**
  The AD account SID for the retrieved account.

- **ADAccountName <string>**
  The AD account name for the retrieved account.

- **Domain <string>**
  The domain for the imported account.

- **State <Citrix.XDInterServiceTypes.ADIdentityState>**
  The state for the account. This can be:
  - **Available**
    The account is not used.
  - **InUse**
    The account is in use.
  - **Error**
    The account is in error (i.e. the account is locked or disabled in AD).
  - **Tainted**
    The account is no longer used, but the password is no longer known.
  - **Lock <Boolean>**
    The account is locked (in the database not in AD).

- **IdentityPoolName <System.String>**
  The name of the containing identity pool.

- **IdentityPoolUid <System.Guid>**
  The GUID identifying the containing identity pool.

**Notes**
In the case of failure the following errors can result.

Error Codes
-----------

PartialData
Only a subset of the available data was returned.

CouldNotQueryDatabase
The query required to get the database was not defined.

PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error

CommunicationError
An error occurred while communicating with the service.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
C:\>Get-AcctADAccount
Return all the AD accounts that are registered in the AD Identity Service.

-------------------------- EXAMPLE 2 --------------------------
C:\>Get-AcctADAccount -IdentityPoolName MyPool -Lock $false
Return all the AD accounts that are registered in the AD Identity Service in the identity pool called "MyPool" that are also locked.

-------------------------- EXAMPLE 3 --------------------------
C:\>Get-AcctADAccount -Filter (IdentityPoolName -Like "p*" -or IdentityPoolName -eq "MyPool")
Return all the AD accounts that are registered in the AD Identity Service in the identity pool called "MyPool" or in an identity pool that has a name that starts with a 'p'. For full details of the advanced filtering aspects of this command see about_Acct_Filtering.
Get-AcctDBConnection

Oct 10, 2013

Gets the database string for the specified data store used by the AdIdentity Service.

Syntax

Get-AcctDBConnection [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns the database connection string for the specified data store.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

Related topics

Get-AcctServiceStatus
Set-AcctDBConnection
Test-AcctDBConnection

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

system.string

The database connection string configured for the AdIdentity Service.

Notes

If the command fails, the following errors can be returned.

Error Codes

---------

NoDBConnections
The database connection string for the AdIdentity Service has not been specified.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AcctDBConnection

Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True
Get the database connection string for the AdIdentity Service.
Get-AcctDBSchema

Oct 10, 2013
Gets a script that creates the AdIdentity Service database schema for the specified data store.

Syntax

Get-AcctDBSchema [-DatabaseName <String>] [-ServiceGroupName <String>] [-ScriptType <ScriptTypes>] [-LocalDatabase] [-Sid <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Gets SQL scripts that can be used to create a new AdIdentity Service database schema, add a new AdIdentity Service to an existing site, remove a AdIdentity Service from a site, or create a database server logon for a AdIdentity Service. If no Sid parameter is provided, the scripts obtained relate to the currently selected AdIdentity Service instance, otherwise the scripts relate to AdIdentity Service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied. The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to a AdIdentity SDK cmdlet. The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured. The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SMDCMD mode'. The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to AdIdentity Service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to AdIdentity Service roles

If ScriptType is Evict, the returned script contains:

- Removal of AdIdentity Service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

If the service uses two data stores they can exist in the same database. You do not need to configure a database before using this command.
Related topics

Set-AcctDBConnection

Parameters

- **DatabaseName**<String>

  Specifies the name of the database for which the schema will be generated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

- **ServiceGroupName**<String>

  Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the AdIdentity services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

- **ScriptType**<ScriptTypes>

  Specifies the type of database script returned. Available script types are:

  **Database**

  Returns a full database script that can be used to create a database schema for the AdIdentity Service in a database instance that does not already contain a schema for this service. The DatabaseName and ServiceGroupName parameters must be specified to create a script of this type.

  **Instance**

  Returns a permissions script that can be used to add further AdIdentity services to an existing database instance that already contains the full AdIdentity service schema, associating the services to the Service Group. The Sid parameter can optionally be specified to create a script of this type.

  **Login**

  Returns a database logon script that can be used to add the required logon accounts to an existing database instance that contains the AdIdentity Service schema. This is used primarily when creating a mirrored database environment. The DatabaseName parameter must be specified to create a script of this type.

  **Evict**
Returns a script that can be used to remove the specified AdIdentity Service from the database entirely. The DatabaseName and Sid parameters must be specified to create a script of this type.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Database</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LocalDatabase** *(SwitchParameter)*

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for AdIdentity services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Sid** *(String)*

Specifies the SID of the controller on which the AdIdentity Service instance to remove from the database is running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **AdminAddress** *(String)*

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**

- **System.String**

A string containing the required SQL script for application to a database.

**Notes**
The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:
- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:
- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned.

Error Codes
-----------

GetSchemasFailed
The database schema could not be found.

ActiveDirectoryAccountResolutionFailed
The specified Active Directory account or Group could not be found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.
AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-AcctDBSchema -DatabaseName MyDB -ServiceGroupName MyServiceGroup > c:\AcctSchema.sql
Get the full database schema for site data store of the AdIdentity Service and copy it to a file called 'c:\AcctSchema.sql'.

This script can then be used to create the schema in a pre-existing database named 'MyDB' that does not already contain a AdIdentity Service site schema.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Get-AcctDBSchema -DatabaseName MyDB -scriptType Login > c:\AdIdentityLogins.sql
Get the logon scripts for the AdIdentity Service.
Get-AcctDBVersionChangeScript

Oct 10, 2013
Gets a script that updates the AdIdentity Service database schema.

Syntax

Get-AcctDBVersionChangeScript -DatabaseName <String> -TargetVersion <Version> [-AdminAddress <String>] [CommonParameters>

Detailed Description

Returns a database script that can be used to upgrade or downgrade the site or secondary schema for the AdIdentity Service from the current schema version to a different version.

Related topics

Get-AcctInstalledDBVersion

Parameters

-DatabaseName <String>

Specifies the name of the database instance to which the update applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-TargetVersion <Version>

Specifies the version of the database you want to update to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values

System.Management.Automation.PSObject

A PSObject containing the required SQL script for application to a database.

Notes

The PSObject returned by this cmdlet contains the following properties:

-- Script The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

-- NeedExclusiveAccess Indicates whether all services in the service group must be shut down during the update or not.

-- CanUndo Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any AdIdentity services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-AcctServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned.

Error Codes

----------

NoOp

The operation was successful but had no effect.

NoDBCConnections

The database connection string for the AdIdentity Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $update = Get-AcctDBVersionChangeScript -DatabaseName MyDb -TargetVersion 1.0.75.0
C:\PS> $update.Script > update_75.sql

Gets an SQL update script to update the current schema to version 1.0.75.0. The resulting update_75.sql script is suitable for direct use with the SQL Server SQLCMD utility.
Get-AcctIdentityPool

Oct 10, 2013

Gets existing identity pools.

Syntax


Detailed Description

Provides the ability to locate existing identity pools.

Related topics

New-AcctIdentityPool
Remove-AcctIdentityPool
Rename-AcctIdentityPool
Set-AcctIdentityPool

Parameters

-**IdentityPoolName**: <String>

  The name of the identity pool.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-**IdentityPoolUid**: <Guid>

  The unique identifier for the identity pool.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Lock**: <Boolean>

  Whether the identity pool is locked or not.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>false</td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td><strong>Default Value</strong></td>
<td><strong>Accept Pipeline Input?</strong></td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td><strong>-Scopeld</strong></td>
<td><code>&lt;Guid&gt;</code></td>
<td></td>
</tr>
<tr>
<td>Gets only results with a scope matching the specified scope identifier.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td><strong>Default Value</strong></td>
<td><strong>Accept Pipeline Input?</strong></td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td><strong>-ScopeName</strong></td>
<td><code>&lt;String&gt;</code></td>
<td></td>
</tr>
<tr>
<td>Gets only results with a scope matching the specified scope name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td><strong>Default Value</strong></td>
<td><strong>Accept Pipeline Input?</strong></td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td><strong>-ReturnTotalRecordCount</strong></td>
<td><code>&lt;SwitchParameter&gt;</code></td>
<td>See about <code>_Acct_Filtering</code> for details.</td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td><strong>Default Value</strong></td>
<td><strong>Accept Pipeline Input?</strong></td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td><strong>-MaxRecordCount</strong></td>
<td><code>&lt;Int32&gt;</code></td>
<td>See about <code>_Acct_Filtering</code> for details.</td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td><strong>Default Value</strong></td>
<td><strong>Accept Pipeline Input?</strong></td>
</tr>
<tr>
<td>false</td>
<td>250</td>
<td>false</td>
</tr>
<tr>
<td><strong>-Skip</strong></td>
<td><code>&lt;Int32&gt;</code></td>
<td></td>
</tr>
<tr>
<td>See about <code>_Acct_Filtering</code> for details.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Required?</td>
<td>Default Value</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------</td>
<td>----------------</td>
</tr>
<tr>
<td>-SortBy</td>
<td>false</td>
<td>0</td>
</tr>
</tbody>
</table>

- **-SortBy**<String>

See about _Acct_Filtering_ for details.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Filter</td>
<td>false</td>
<td>0</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Filter**<String>

See about _Acct_Filtering_ for details.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-AdminAddress</td>
<td>false</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value will become the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

**Return Values**

**Citrix.AdIdentity.Sdk.IdentityPool**

This object provides details of the identity pool and contains the following information:

- **IdentityPoolName**<string>
  
  The name of the identity pool.

- **IdentityPoolUid**<Guid>
  
  The unique identifier for the identity pool.
NamingScheme <string>
The naming scheme for the identity pool.

NamingSchemeType <Citrix.XDInterServiceTypes.ADIdentityNamingScheme>
The naming scheme type for the identity pool. This can be one of the following:
Numeric - naming scheme uses numeric indexes
Alphabetic - naming scheme uses alphabetic indexes

StartCount <int>
The next index to be used when creating an identity from the identity pool.

OU <string>
The Active Directory distinguished name for the OU in which accounts created from this identity pool will be created.

Domain <string>
The Active Directory domain that accounts in the pool belong to.

Lock <Boolean>
Indicates if the identity pool is locked.

Notes
In the case of failure, the following errors can result.

Error Codes
-------------
PartialData
Only a subset of the available data was returned.
CouldNotQueryDatabase
The query required to get the database was not defined.
PermissionDenied
The user does not have administrative rights to perform this operation.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error
CommunicationError
An error occurred while communicating with the service.
The operation could not be completed because the database for the service is not configured.

A filtering expression was supplied that could not be interpreted for this cmdlet.

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Get-AcctIdentityPool

IdentityPoolName : MyPool
IdentityPoolUid  : 22072d9e-6a8f-494b-a5bc-2ef18ca4b915
NamingScheme     : Acc####
NamingSchemeType : Numeric
StartCount       : 1
OU               :
Domain           : mydomain.com
Lock             : True

IdentityPoolName : MyPool2
IdentityPoolUid  : 03743136-e43b-4a87-af74-ab71686b3c16
NamingScheme     : Test####
NamingSchemeType : Alphabetic
StartCount       : 1
OU               :
Domain           : mydomain.com
Lock             : False

Gets all the identity pools.

-------------------------- EXAMPLE 2 --------------------------

C:\PS>Get-AcctIdentityPool -IdentityPoolName M*

IdentityPoolName : MyPool
IdentityPoolUid  : 22072d9e-6a8f-494b-a5bc-2ef18ca4b915
NamingScheme     : Acc####
NamingSchemeType : Numeric
StartCount       : 1
OU               :
Domain           : mydomain.com
Lock             : True

Gets all the identity pools beginning with the character 'M'.

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.519
Get-AcctInstalledDBVersion

Oct 10, 2013

Gets a list of all available database schema versions for the AdIdentity Service.

Syntax

Get-AcctInstalledDBVersion [-Upgrade] [-Downgrade] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns the current version of the AdIdentity Service database schema, if no flags are set, otherwise returns versions for which upgrade or downgrade scripts are available and have been stored in the database.

Related topics

Parameters

- **-Upgrade** <SwitchParameter>

  Specifies that only schema versions to which the current database version can be updated should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

  | Accept Pipeline Input? | false |

- **-Downgrade** <SwitchParameter>

  Specifies that only schema versions to which the current database version can be reverted should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

  | Accept Pipeline Input? | false |

- **-AdminAddress** <String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

  | Accept Pipeline Input? | false |

Return Values
System\Version

The Get-AcctInstalledDbVersion command returns objects containing the new definition of the AdIdentity Service database schema version.

Major <Integer>
Minor <Integer>
Build <Integer>
Revision <Integer>

Notes
If the command fails, the following errors can be returned.

Error Codes
----------

InvalidParameterCombination
Both the Upgrade and Downgrade flags were specified.

NoOp
The operation was successful but had no effect.

NoDBConnections
The database connection string for the AdIdentity Service has not been specified.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AcctInstalledDBVersion

<table>
<thead>
<tr>
<th>Major</th>
<th>Minor</th>
<th>Build</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Get the currently installed version of the AdIdentity Service database schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-AcctInstalledDBVersion -Upgrade

<table>
<thead>
<tr>
<th>Major</th>
<th>Minor</th>
<th>Build</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Get the versions of the AdIdentity Service database schema for which upgrade scripts are supplied.
Get-AcctScopedObject

Oct 10, 2013

Gets the details of the scoped objects for the AdIdentity Service.

Syntax


Detailed Description

Returns a list of directly scoped objects including the names and identifiers of both the scope and object as well as the object description for display purposes.

There will be at least one result for every directly scoped object. When an object is associated with multiple scopes the output contains one result per scope duplicating the object details.

No records are returned for the All scope, though if an object is not in any scope a result with a null ScopeId and ScopeName will be returned.

Related topics

Parameters

- **-ScopeId**<Guid>

Gets scoped object entries for the given scope identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-ScopeName**<String>

Gets scoped object entries with the given scope name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-ObjectType**<ScopedObjectType>

Gets scoped object entries for objects of the given type.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.524
<table>
<thead>
<tr>
<th><strong>-ObjectId</strong>&lt;String&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets scoped object entries for objects with the specified object identifier.</td>
<td></td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td>false</td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-ObjectName</strong>&lt;String&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets scoped object entries for objects with the specified object identifier.</td>
<td></td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td>false</td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-Description</strong>&lt;String&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets scoped object entries for objects with the specified description.</td>
<td></td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td>false</td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-Property</strong>&lt;String[]&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.</td>
<td></td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td>false</td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-ReturnTotalRecordCount</strong>&lt;SwitchParameter&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Acct_Filtering for details.</td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Required?</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>MaxRecordCount (int32)</td>
<td>false</td>
</tr>
<tr>
<td>Specifies the maximum number of records to return.</td>
<td></td>
</tr>
<tr>
<td>Skip (int32)</td>
<td>false</td>
</tr>
<tr>
<td>Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.</td>
<td></td>
</tr>
<tr>
<td>SortBy (string)</td>
<td>false</td>
</tr>
<tr>
<td>Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.</td>
<td></td>
</tr>
<tr>
<td>Filter (string)</td>
<td>false</td>
</tr>
<tr>
<td>Gets records that match a PowerShell-style filter expression. See about_Acct_Filtering for details.</td>
<td></td>
</tr>
</tbody>
</table>
**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**

Citrix.AdIdentity.Sdk.ScopedObject

The Get-AcctScopedObject command returns an object containing the following properties:

- **ScopeId <Guid?>**
  - Specifies the unique identifier of the scope.

- **ScopeName <String>**
  - Specifies the display name of the scope.

- **ObjectType <ScopedObjectType>**
  - Type of the object this entry relates to.

- **ObjectId <String>**
  - Unique identifier of the object.

- **ObjectName <String>**
  - Display name of the object

- **Description <String>**
  - Description of the object (possibly $null if the object type does not have a description).

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

----------

UnknownObject

One of the specified objects was not found.

PartialData
Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AcctScopedObject -ObjectType Scheme

ScopeId : eff6f464-f1ee-4442-add3-99982e0cec01
ScopeName : Sales
ObjectType : Scheme
ObjectId : cd4174ee-9e4b-4e57-b126-9dbf757fe493
ObjectName : MyExampleScheme
Description : Test scheme
ScopeId : 304e0fa7-d390-47f0-a94f-7e956a324c41
ScopeName : Finance
ObjectType : Scheme
ObjectId    : cd4174ee-9e4b-4e57-b126-9dbf757fe493
ObjectName  : MyExampleScheme
Description : Test scheme

ScopeId     :
ScopeName   :
ObjectType  : Scheme
ObjectId    : 5062e46b-71bc-4ac9-901a-30fe6797e2f6
ObjectName  : AnotherScheme
Description : Another scheme in no scopes

Gets all of the scoped objects with type Scheme. The example output shows a scheme object (MyExampleScheme) in two scopes Sales and Finance, and another scheme (AnotherScheme) that is not in any scope. The ScopeId and ScopeName values returned are null in the final record.
Get-AcctService

Oct 10, 2013
Gets the service record entries for the AdIdentity Service.

Syntax


Detailed Description

Returns instances of the AdIdentity Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Related topics

Parameters

-Property <String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ReturnTotalRecordCount <SwitchParameter>

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Acct_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-MaxRecordCount <Int32>

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
</tbody>
</table>
### -Skip<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -SortBy<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -Filter<String>

 Gets records that match a PowerShell-style filter expression. See about_Acct_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.AdlIdentity.Sdk.Service

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.531
The Get-AcctServiceInstance command returns an object containing the following properties.

Uid <Integer>
Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>
Specifies the unique identifier for the service instance.

DNSName <String>
Specifies the domain name of the host on which the service runs.

MachineName <String>
Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>
Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>
Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>
Specifies the date and time at which the service was last stopped or restarted.

OSType
Specifies the operating system installed on the host on which the service runs.

OSVersion
Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion
Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>
Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>
Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>
Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.
Notes

If the command fails, the following errors can be returned.

Error Codes
----------

UnknownObject
One of the specified objects was not found.

PartialData
Only a subset of the available data was returned.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase
The query required to get the database was not defined.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-AcctService

Uid : 1
ServiceHostId : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName : MyServer.company.com
MachineName : MYSERVER
CurrentState : On
LastStartTime : 04/04/2011 15:25:38
LastActivityTime : 04/04/2011 15:33:39
OSType : Win32NT
OSVersion : 6.1.7600.0
ServiceVersion : 5.1.0.0
DatabaseUserName : NT AUTHORITY\NETWORK SERVICE
SID : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {MySiteService1, MySiteService2...}

Get all the instances of the AdIdentity Service running in the current service group.
Get-AcctServiceAddedCapability

Oct 10, 2013
Gets any added capabilities for the AdIdentity Service on the controller.

Syntax
Get-AcctServiceAddedCapability [-AdminAddress <String>] [CommonParameters>

Detailed Description
Enables updates to the AdIdentity Service on the controller to be detected.
You do not need to configure a database connection before using this command.

Related topics
Parameters
-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
System.String
String containing added capabilities.

Notes
If the command fails, the following errors can be returned.

Error Codes
---------
DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

`PermissionDenied`

You do not have permission to execute this command.

`AuthorizationError`

There was a problem communicating with the Citrix Delegated Administration Service.

`CommunicationError`

There was a problem communicating with the remote service.

`ExceptionThrown`

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AcctServiceAddedCapability
Get the added capabilities of the AdIdentity Service.
Get-AcctServiceInstance

Oct 10, 2013
Gets the service instance entries for the AdIdentity Service.

Syntax

Get-AcctServiceInstance [-AdminAddress <String>] [CommonParameters]

Detailed Description

Returns service interfaces published by the instance of the AdIdentity Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Related topics

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

Return Values

Citrix.AdIdentity.Sdk.ServiceInstance

The Get-AcctServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.
ServiceType <String>
Specifies the service instance type. For this service, the service instance type is always Acct.

Address
Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding
Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version
Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>
Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>
Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>
Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

- SDK - for PowerShell operations
- InterService - for operations between different services
- Peer - for communications between services of the same type

Metadata <Citrix.AdIdentity.Sdk.Metadata[]>
The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes
If the command fails, the following errors can be returned.

Error Codes
---------

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

--------------- EXAMPLE 1 --------------------------

c:\PS>Get-AcctServiceInstance

Address            : http://MyServer.com:80/Citrix/AdIdentityService
Binding            : wcf_HTTP_kerb
InterfaceType      : SDK
Metadata           :
MetadataMap        :
ServiceAccount     : ENG\MyAccount$
ServiceAccountSid  : S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName   : MyServiceGroup
ServiceGroupUid    : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceUid : 00000000-0000-0000-0000-000000000000
ServiceType        : Acct
Version            : 1

Address            : http://MyServer.com:80/Citrix/AdIdentityService/IServiceApi
Binding            : wcf_HTTP_kerb
InterfaceType      : InterService
Metadata           :
MetadataMap        :
ServiceAccount     : ENG\MyAccount
ServiceAccountSid  : S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName : MyServiceGroup
ServiceGroupId : a88d2f6b-c00a-4f76-9c38-e8330093b54c
ServiceInstanceId : 00000000-0000-0000-0000-000000000000
ServiceType : Acct
Version : 1

Get all instances of the AdIdentity Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.
Get-AcctServiceStatus

Oct 10, 2013
Gets the current status of the AdIdentity Service on the controller.

Syntax

Get-AcctServiceStatus [-AdminAddress <String>] [CommonParameters>

Detailed Description

Enables the status of the AdIdentity Service on the controller to be monitored. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured.

Related topics

Set-AcctDBConnection
Test-AcctDBConnection
Get-AcctDBConnection
Get-AcctDBSchema

Parameters

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true</td>
</tr>
</tbody>
</table>

Return Values


The Get-AcctServiceStatus command returns an object containing the status of the AdIdentity Service together with extra diagnostics information.

DBUnconfigured

The AdIdentity Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the AdIdentity Service. This may be because the service attempted to log
on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**

The expected stored procedures are missing from the database. This may be because the AdIdentity Service schema has not been added to the database.

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**

The version of the AdIdentity Service currently in use is incompatible with the version of the AdIdentity Service schema on the database. Upgrade the AdIdentity Service to a more recent version.

**DBOlderVersionThanService**

The version of the AdIdentity Service schema on the database is incompatible with the version of the AdIdentity Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**

A database schema upgrade is currently in progress.

**OK**

The AdIdentity Service is running and is connected to a database containing a valid schema.

**Failed**

The AdIdentity Service has failed.

**Unknown**

(0) The service status cannot be determined.

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

----------------

**DatabaseError**

An error occurred in the service while attempting a database operation.

**DatabaseNotConfigured**

The operation could not be completed because the database for the service is not configured.

**DataStoreException**

An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AcctServiceStatus

DBUnconfigured
Get the current status of the AdIdentity Service.
New-AcctADAccount

Oct 10, 2013
Creates AD computer accounts in the specified identity pool.

Syntax

New-AcctADAccount [-IdentityPoolName] <String> -Count <Int32> [-StartCount <Int32>] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

New-AcctADAccount [-IdentityPoolName] <String> -ADAccountName <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

New-AcctADAccount -IdentityPoolUid <Guid> -Count <Int32> [-StartCount <Int32>] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

New-AcctADAccount -IdentityPoolUid <Guid> -ADAccountName <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Provides the ability to create new AD computer accounts and register them in an already existing identity pool.

The accounts are created using the information stored in the identity pool. This provides the account name (via the Naming Scheme property and Start Count), domain, and OU.

The runspace used for this command must have sufficient privileges in Active Directory to create the new computer accounts.

The AD account names will pad the index to use all the space specified in the identity pool naming scheme (e.g. "acc###" will become "acc001"). However, if the index overflows the available space the cmdlet expands the format to use the next incremental number (e.g. "acc###" will become "acc1000" if the index is 10000, which cannot fit into the three '#' placeholders). If this expanded name exceeds the 15 character name limit, the accounts are not created.

There can be only one creation process running for a specific identity pool at any one time. Attempting to start another account creation process while an existing one is executing results in an error being returned.

Related topics

Add-AcctADAccount
Remove-AcctADAccount
Get-AcctADAccount
Repair-AcctADAccount
Unlock-AcctADAccount
Update-AcctADAccount

Parameters
**-IdentityPoolName**<String>

The name of the identity pool in which to create the accounts.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

**-Count**<Int32>

The number of accounts to create.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

**-ADAccountName**<String[]>

The AD account names to be created. These are just the simple machine account names e.g. MyVM001

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

**-IdentityPoolUid**<Guid>

The unique identifier for the identity pool in which the accounts will be created.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

**-StartCount**<Int32>

The start index for the create process.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |
-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | LocalHost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values

Citrix.AdIdentity.Sdk.AccountOperationDetailedSummary

The Add-AcctADAccount returns an object that contains the following parameters;

SuccessfulAccountsCount <int>
The number of accounts that were added successfully

FailedAccountsCount <int>
The number of accounts that were not added.

FailedAccounts <Citrix.AdIdentity.Sdk.AccountError[]>
The list of accounts that failed to be added. Each one has the following parameters;

ADAccountName <string>

ADAccountSid <String>

ErrorReason <string>

This can be one of the following

IdentityDuplicateObjectExists

An identity with the same SID already exists.
ADServiceDatabaseError
An error occurred in the service while attempting a database operation.

ADServiceDatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ADServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

FailedToConnectToDomainController
Contacting Active Directory failed.

FailedToGetOrganizationUnitInAD
Failed to access the OU in Active Directory.

FailedToGetDefaultComputerContainerInAD
Failed to access the default computers container in Active Directory.

FailedToCreateComputerAccountInAD
Failed to create the computer account in Active Directory.

FailedToAccessComputerAccountInAD
Failed to read the newly created computer account in Active Directory.

FailedToGetSidFromAD
Failed to get the SID for the created account from Active Directory.

FailedToSetSamAccountNameInAD
Failed to set the SAM account name in Active Directory for the account created.

FailedToSetUserAccountControlInAD
Failed to set the user account controller properties for the account created in Active Directory.

FailedToSaveChangeInAD
Failed to save the changes made to the created computer account in Active Directory.

FailedToSetPasswordInAD
Failed to set the password for the created computer account in Active Directory.

FailedToEnableAccountInAD
Failed to enable the newly created computer account in Active Directory.

ComputerNameAlreadyInUseInAD

The computer name for the computer to create is in use in Active Directory.

FailedToGetDistinguishedNameInAD

Failed to get the distinguished name for the created computer account in ActiveDirectory.

FailedToSetDnsHostNameInAD

Failed to set the Dns Host Name property for the created computer account in ActiveDirectory.

FailedToSetDisplayNameInAD

Failed to set the DisplayName property for the created computer account in ActiveDirectory.

FailedToWriteServicePrincipalNameInAD

Failed to set the ServicePrincipalName property for the created computer account in ActiveDirectory.

DiagnosticInformation <Exception>

Any other error information

SuccessfulAccounts <Citrix.AdIdentity.Sdk.Identity[]>

The list of accounts that were successfully added. Each object provides details of the identity and contains the following information:

ADAccountSID <string>

The Sid of the identity.

ADAccountName <string>

The account name for the identity.

Domain <string>

The domain name that the account was created in.

State <string>

The current state of the AD account. This can be one of the following:

Error

The account is locked or disabled in AD.

Available

The account is in AD and available to be consumed by the other Machine Creation Services.
InUse
The account is in AD and is being consumed by the other Machine Creation Services.

Tainted
The account is in AD and no longer consumed by other Machine Creation Services. However, the password is no longer known so cannot be reused without 'Repairing' the account. See repair-AcctADAccount for details.

Lock <Boolean>
Indicates if the identity pool is locked.

Notes
In the case of failure, the following errors can result.

Error Codes

----------
NamingSchemeNotSpecifiedForIdentityPool
No naming scheme is defined in the specified identity pool.
IdentityPoolObjectNotFound
The specified identity pool was not located.
IdentityPoolAlreadyLocked
The specified identity pool is locked.
PermissionDenied
The user does not have administrative rights to perform this operation.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error
DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.
CommunicationError
An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> New-AcctADAccount -IdentityPoolName MyPool -Count 2 -OutVariable result

SuccessfulAccounts                SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
------------------                -----------------------       -------------------    --------------
{MyDomain\ACC001, MyDomain\ACC002} 2                            0                      {}

$result[0].SuccessfulAccounts

ADAccountSid : S-1-5-21-1315084875-1285793635-2418178940-2684
ADAccountName : MyDomain\ACC001
Domain : MyDomain.com
State : Available
Lock : False

ADAccountSid : S-1-5-21-1315084875-1285793635-2418178940-2685
ADAccountName : MyDomain\ACC002
Domain : MyDomain.com
State : Available
Lock : False

Creates two new AD accounts and registers them in the identity pool called "MyPool".

-------------------------- EXAMPLE 2 --------------------------

c:\PS> New-AcctADAccount -IdentityPoolName MyPool -Count 2 -StartCount 50 -OutVariable result

SuccessfulAccounts                SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
------------------                -----------------------       -------------------    --------------
{MyDomain\ACC050, MyDomain\ACC051} 2                            0                      {}

$result[0].SuccessfulAccounts

ADAccountSid : S-1-5-21-1315084875-1285793635-2418178940-2686
ADAccountName : MyDomain\ACC050
Domain : MyDomain.com
State : Available
Lock : False

ADAccountSid : S-1-5-21-1315084875-1285793635-2418178940-2687
ADAccountName : MyDomain\ACC051
Domain : MyDomain.com
State : Available
Lock : False

Creates two new AD accounts and registers them in the identity pool called "MyPool", starting from an index of 50.
New-AcctIdentityPool

Oct 10, 2013

Creates a new identity pool.

Syntax


Detailed Description

Provides the ability to create identity pools that can be used to store AD computer accounts.

The naming scheme, naming scheme type, and domain must be specified if the identity pool is to be used to create new accounts.

Each identity pool is tied to a single domain. All the identities in an identity pool belong to the same domain. If the domain is not specified by a parameter to this command the domain will be set when an account is imported into it.

Related topics

Remove-AcctIdentityPool
Rename-AcctIdentityPool
Set-AcctIdentityPool
Test-AcctIdentityPoolNameAvailable
New-AcctADAccount

Parameters

-IdentityPoolName <String>
The name of the identity pool. This must not contain any of the following characters \/:#.*?=<>|}{'"'

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-NamingScheme <String>

Defines the template name for AD accounts created in the identity pool. The scheme can consist of fixed characters and a variable part defined by '#' characters. There can be only one variable region defined. The number of '#' characters defines the minimum length of the variable region. For example, a naming scheme of H#### could create accounts called H0001, H0002 (for a numeric scheme type) or HAAAA, HAAAB (for an alphabetic type).

Citrix recommends that you define a naming scheme that will not clash with accounts which already exist in AD; account creation will fail if a clash occurs.

There are restrictions on what constitutes a valid computer name: Minimum name length: 2 (DNS) Maximum name length: 15 bytes (NetBIOS) The following characters are not allowed in a computer name: backslash () slash mark (/) colon (:) asterisk (*) question mark (?) quotation mark (") less than sign (<) greater than sign (>) vertical bar (|) comma (,) tilde (~) exclamation point (!) at sign (@) number sign (#) dollar sign ($) percent (%) caret (^) ampersand (&) apostrophe (') parenthesis (()) braces ([]) underscore (_) The following names are reserved

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.552
and must not be used at the end of the naming scheme: -GATEWAY -GW -TAC Names must not start with a period (NetBIOS). Names must not be composed entirely of numbers (NetBIOS). Names must not contain a blank or space characters (DNS).

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**NamingSchemeType**<ADIdentityNamingScheme>

The type of naming scheme. This can be Numeric or Alphabetic. This defines the format of the variable part of the AD account names that will be created.

| Required? | false |
| Default Value | Numeric |
| Accept Pipeline Input? | false |

**OU**<String>

The OU that computer accounts will be created into. If this is not specified, accounts are created into the default account container specified by AD. This is the 'Computers' container for out-of-the-box installations of AD. The OU must be a valid AD container and of the domain specified for the pool.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**Domain**<String>

The AD domain name for the pool. Specify this in FQDN format; for example, MyDomain.com.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**AllowUnicode**<SwitchParameter>

Allow the naming scheme to have characters other than alphanumeric characters.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**StartCount**<Int32>

Defines the next number that will be used if creating new AD accounts for the identity pool.
### -Scope<String[]>

The administration scopes to be applied to the new identity pool.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -LoggingId<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### Return Values

Citrix.AdIdentity.Sdk.IdentityPool

This object provides details of the identity pool and contains the following information:

- **IdentityPoolName <string>**
  - The name of the identity pool.

- **IdentityPoolUid <Guid>**
  - The unique identifier for the identity pool.

- **NamingScheme <string>**
  - The naming scheme for the identity pool.
NamingSchemeType <Citrix.XDInterServiceTypes.ADIdentityNamingScheme>
The naming scheme type for the identity pool. This can be one of the following:

Numeric - naming scheme uses numeric indexes
Alphabetic - naming scheme uses alphabetic indexes

StartCount <int>
The next index to be used when creating an identity from the identity pool.

OU <string>
The Active Directory distinguished name for the OU in which accounts created from this identity pool will be created.

Domain <string>
The Active Directory domain that accounts in the pool belong to.

Lock <Boolean>
Indicates if the identity pool is locked.

Notes
In the case of failure, the following errors can result.

Error Codes
----------
UnableToConvertDomainName
Unable to convert domain name to DNS format.
NamingSchemeNotEnoughCharacters
Naming scheme does not have enough characters specified.
NamingSchemeTooManyCharacters
Naming scheme has too many characters specified.
NamingSchemeIllegalCharacter
Naming scheme contains illegal characters.
NamingSchemeMayNotStartWithPeriod
Naming scheme starts with a period (.) character.
NamingSchemeMayNotBeAllNumbers
Naming scheme contains only numbers.
NamingSchemeMissingNumericSpecifications
Naming scheme does not contain any variable specification (i.e. no '#' characters are specified).
NamingSchemeHasMoreThanOneSetOfHashes
Naming scheme has more than one variable region (i.e. there are '#' characters separated by other characters).
IdentityPoolDuplicateObjectExists
An identity pool with the same name already exists.

IdentityPoolOUInvalid
Identity Pool OU invalid as it does not exist.

IdentityPoolOUOfWrongDomain
IdentityPool OU invalid as it refers to a different domain to the domain specified for the pool.

InvalidIdentityPoolParameterCombination
Caused by either of the following validation errors:

* If an OU is specified then a domain must also be specified.

* NamingScheme, NamingSchemeType and Domain must all be present if any of these are specified.

NamingSchemeIllegalComputerName
The naming scheme supplied is not valid.

PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS> New-AcctIdentityPool -IdentityPoolName MyPool -NamingScheme Acc#### -Domain MyDomain.com -NamingSchemeType Numeric
IdentityPoolName : MyPool
IdentityPoolUid : 22072d9e-6a8f-494b-a5bc-2ef18ca4b915
NamingScheme      : Acc####
NamingSchemeType  : Numeric
StartCount        : 1
OU                :
Domain            : MyDomain.com
Lock              : False

Create a new identity pool from which accounts can be created in the domain called "MyDomain.com" using a numeric scheme of the format Acc####. The first account that will be created from this identity pool definition is Acc0001.

New AD accounts can be imported into this pool too, but must be from the Domain "MyDomain.com".
Remove-AcctADAccount

Oct 10, 2013
Removes AD computer accounts from an identity pool.

Syntax


Remove-AcctADAccount [-IdentityPoolName] <String> -ADAccountName <String[]> [-RemovalOption <ADIdentityRemoveAccountOption>] [-Force] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AcctADAccount -IdentityPoolUid <Guid> -ADAccountName <String[]> [-RemovalOption <ADIdentityRemoveAccountOption>] [-Force] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AcctADAccount -IdentityPoolUid <Guid> -ADAccountSid <String[]> [-RemovalOption <ADIdentityRemoveAccountOption>] [-Force] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove AD accounts from an identity pool. This removes the AD account from the Citrix service management scope. This process provides the options for removing the account in AD (or disabling it) if required.

All aspects of this command that need to make modifications to the accounts in AD will use the account that the runspace is using. This means that if an account is to be removed from AD or disabled, the user performing the operation in PowerShell must have sufficient privileges in AD for this operation to complete successfully.

If the option to remove the account from AD or to disable it in AD is specified, the AD operation must succeed for the account to be removed from the Citrix AD Identity Service database. Use caution when using the Force parameter because this allows removal of accounts that are in the 'inUse' state, which may result in the machines becoming unusable.

Related topics

New-AcctADAccount
Add-AcctADAccount
Repair-AcctADAccount
Unlock-AcctADAccount
Update-AcctADAccount
Get-AcctADAccount

Parameters

- **-IdentityPoolName** <String>

The identity pool that the accounts are to be removed from.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
### -ADAccountSid
The Active Directory Account SID for the account to be removed.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

### -ADAccountName
The AD account name to be removed. AD accounts are accepted in the following formats: Fully qualified DN e.g. CN=MyComputer,OU=Computers,DC=MyDomain,DC=Com; UPN format e.g MyComputer@MyDomain.Com; Domain qualified e.g MyDomain\MyComputer.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

### -IdentityPoolUid
The unique identifier for the identity pool that the accounts are to be removed from.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

### -RemovalOption
Defines the behavior relating to the AD account.

| Required? | false |
| Default Value | None |
| Accept Pipeline Input? | false |

### -Force
Indicates if accounts that are marked as 'in-use' can be removed.

| Required? | false |
| Default Value | |
Accept Pipeline Input? | false
---|---
-**LoggingId**<Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-**AdminAddress**<String>
Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

Return Values
CitrixAdIdentity.Sdk.AccountOperationSummary
The remove-AcctADAccout command returns an object with the following parameters:
SuccessfulAccountsCount <int>
The number of accounts that were removed successfully.
FailedAccountsCount <int>
The number of accounts that were not removed.
FailedAccounts <CitrixAdIdentity.Sdk.AccountError[]>
The list of accounts that failed to be removed. Each one has the following parameters:
ADAccountName <string>
ADAccountSid <String>
ErrorReason <AdIdentityStatus>
This can be one of the following
UnableToConvertDomain
IdentityNotLocatedInDomain
IdentityNotInIdentityPool
IdentityObjectInUse
IdentityObjectLocked
ADServiceDatabaseError
ADServiceDatabaseNotConfigured
ADServiceStatusInvalidDb
FailedToConnectToDomainController
FailedToDisableAccountInAD
FailedToDeleteAccountInAD
FailedToExecuteSearchInAD
FailedToAccessComputerAccountInAD
DiagnosticInformation <Exception>
Any other error information

Notes
In the case of failure, the following errors can result.

Error Codes
----------

IdentityPoolNotFound
The specified identity pool was not found.
IdentityPoolAlreadyLocked
The specified identity pool was locked by another operation.
PermissionDenied
The user does not have administrative rights to perform this operation.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.
DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for
for various reasons.
CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

----------- EXAMPLE 1 -----------

C:\PS>Remove-AcctADAccount -IdentityPool MyPool -ADAccountName "Domain\account","domain\account2"

---

SuccessfulAccountsCount FailedAccountsCount FailedAccounts
----------------------- ------------------- -------------------
2                      0        {}

Removes two accounts (account and account2) from the identity pool called "MyPool", leaving the AD accounts untouched.

----------- EXAMPLE 2 -----------

C:\PS>Remove-AcctADAccount -IdentityPool MyPool -RemovalOption Delete -ADAccountName "Domain\account","domain\account2"

---

SuccessfulAccountsCount FailedAccountsCount FailedAccounts
----------------------- ------------------- -------------------
2                      0        {}

Removes two accounts (account and account2) from the identity pool called "MyPool" (and from Active Directory).

----------- EXAMPLE 3 -----------

C:\PS>Remove-AcctADAccount -IdentityPool MyPool -ADAccountName "Domain\account","domain\account2" -Force

---

SuccessfulAccountsCount FailedAccountsCount FailedAccounts
----------------------- ------------------- -------------------
2                      0        {}

Removes two accounts (account and account2) from the identity pool called "MyPool", leaving the AD accounts untouched. The accounts are removed regardless of whether they are in the 'inUse' state or not.

----------- EXAMPLE 4 -----------

C:\PS>Remove-AcctADAccount -IdentityPool MyPool -ADAccountName "Domain\account","domain\account2" -OutVariable result

---

SuccessfulAccountsCount FailedAccountsCount FailedAccounts
----------------------- ------------------- -------------------
1                      1        {account2}

C:\PS>$result[0].FailedAccounts

ADAccountName       ADAccountSid   ErrorReason
----------------------- ------------------- ---------------
Domain\account2      account2      IdentityObjectLocked
Shows failure of removal of one of two accounts and how to retrieve the failure reason.

-------------------------- EXAMPLE 5 --------------------------

C:\PS>Remove-AcctADAccount -IdentityPool MyPool -ADAccountSid S-1-5-21-1315084875-1285793635-2418178940-2685

<table>
<thead>
<tr>
<th>SuccessfulAccountsCount</th>
<th>FailedAccountsCount</th>
<th>FailedAccounts</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>{}</td>
</tr>
</tbody>
</table>

Removes one account (S-1-5-21-1315084875-1285793635-2418178940-2685) from the identity pool called "MyPool", leaving the AD accounts untouched.
Remove-AcctIdentityPool

Oct 10, 2013
Removes identity pools.

Syntax


Remove-AcctIdentityPool -IdentityPoolUid <Guid> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

Provides the ability to remove identity pools. The identity pool must be emptied of any AD accounts that it contains before it can be removed.

Related topics

New-AcctIdentityPool

Rename-AcctIdentityPool

Set-AcctIdentityPool

Unlock-AcctIdentityPool

Parameters

-**IdentityPoolName**<String>

The name of the identity pool to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**IdentityPoolUid**<Guid>

The unique identifier for the identity pool to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create
high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress** `<String>`

  Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Notes**

In the case of failure the following errors can be produced.

**Error Codes**

<table>
<thead>
<tr>
<th>IdentityPoolObjectNotFound</th>
</tr>
</thead>
<tbody>
<tr>
<td>The specified identity pool could not be located.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UnableToRemoveDueToAssociatedAccounts</th>
</tr>
</thead>
<tbody>
<tr>
<td>The identity pool is not empty.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PermissionDenied</th>
</tr>
</thead>
<tbody>
<tr>
<td>The user does not have administrative rights to perform this operation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ConfigurationLoggingError</th>
</tr>
</thead>
<tbody>
<tr>
<td>The operation could not be performed because of a configuration logging error</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DatabaseError</th>
</tr>
</thead>
<tbody>
<tr>
<td>An error occurred in the service while attempting a database operation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DatabaseNotConfigured</th>
</tr>
</thead>
<tbody>
<tr>
<td>The operation could not be completed because the database for the service is not configured.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ServiceStatusInvalidDb</th>
</tr>
</thead>
</table>
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

---------------------------- EXAMPLE 1 -----------------------------

c:\Remove-AcctIdentityPool -IdentityPoolName MyPool
Removes the identity pool called "MyPool".
Remove-AcctIdentityPoolMetadata

Oct 10, 2013
Removes metadata from the given IdentityPool.

Syntax

Remove-AcctIdentityPoolMetadata [-IdentityPoolUid] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AcctIdentityPoolMetadata [-IdentityPoolUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AcctIdentityPoolMetadata [-IdentityPoolName] <String> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AcctIdentityPoolMetadata [-IdentityPoolName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AcctIdentityPoolMetadata [-InputObject] <IdentityPool[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AcctIdentityPoolMetadata [-InputObject] <IdentityPool[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given IdentityPool.

Related topics

Set-AcctIdentityPoolMetadata

Parameters

- **IdentityPoolUid** <Guid>
  Id of the IdentityPool

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **IdentityPoolName** <String>
  Name of the IdentityPool

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.567
### -InputObject <IdentityPool[]>

Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

### -Name <String>

The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

### -Map <PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

### -LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

Error Codes

------------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AcctIdentityPool | % { Remove-AcctIdentityPoolMetadata -Map $_.MetadataMap }
Remove all metadata from all IdentityPool objects.
Remove-AcctIdentityPoolScope

Oct 10, 2013
Remove the specified IdentityPool(s) from the given scope(s).

Syntax

Remove-AcctIdentityPoolScope [-Scope] <String[]> -InputObject <IdentityPool[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AcctIdentityPoolScope [-Scope] <String[]> -IdentityPoolUid <Guid[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AcctIdentityPoolScope [-Scope] <String[]> -IdentityPoolName <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The RemoveAcctIdentityPoolScope cmdlet is used to remove one or more IdentityPool objects from the given scope(s).

There are multiple parameter sets for this cmdlet, allowing you to identify the IdentityPool objects in different ways:

- IdentityPool objects can be piped in or specified by the InputObject parameter
- The IdentityPoolUid parameter specifies objects by IdentityPoolUid
- The IdentityPoolName parameter specifies objects by IdentityPoolName (supports wildcards)

To remove a IdentityPool from a scope you need permission to change the scopes of the IdentityPool.

If the IdentityPool is not in a specified scope, that scope will be silently ignored.

Related topics

Add-AcctIdentityPoolScope

Get-AcctScopedObject

Parameters

- **-Scope**<String[]>

  Specifies the scopes to remove the objects from.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-InputObject**<IdentityPool[]>

  Specifies the IdentityPool objects to be removed.
<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th><strong>true</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><strong>true (ByValue, ByPropertyName)</strong></td>
</tr>
</tbody>
</table>

**-IdentityPoolUid<Guid[]>**

Specifies the IdentityPool objects to be removed by IdentityPoolUid.

<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th><strong>true</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><strong>true (ByValue, ByPropertyName)</strong></td>
</tr>
</tbody>
</table>

**-IdentityPoolName<String[]>**

Specifies the IdentityPool objects to be removed by IdentityPoolName.

<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th><strong>true</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><strong>true (ByValue, ByPropertyName)</strong></td>
</tr>
</tbody>
</table>

**-LoggingId<Guid>**

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th><strong>false</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><strong>false</strong></td>
</tr>
</tbody>
</table>

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th><strong>false</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td><strong>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</strong></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><strong>false</strong></td>
</tr>
</tbody>
</table>
Return Values
None

Notes
If the command fails, the following errors can be returned.

Error Codes
---------

UnknownObject
One of the specified objects was not found.

ScopeNotFound
One of the specified scopes was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------- EXAMPLE 1 --------------------------
c:\PS>Remove-AcctIdentityPoolScope Finance -IdentityPoolUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
Removes a single IdentityPool from the 'Finance' scope.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Remove-AcctIdentityPoolScope Finance,Marketing -IdentityPoolUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
Removes a single IdentityPool from multiple scopes.

-------------------------- EXAMPLE 3 --------------------------

c:\PS>Get-AcctIdentityPool | Remove-AcctIdentityPoolScope Finance
Removes all visible IdentityPool objects from the 'Finance' scope.

-------------------------- EXAMPLE 4 --------------------------

c:\PS>Remove-AcctIdentityPoolScope Finance -IdentityPoolName A*
Removes IdentityPool objects with a name starting with an 'A' from the 'Finance' scope.
Remove-AcctServiceMetadata

Oct 10, 2013
Removes metadata from the given Service.

Syntax

Remove-AcctServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AcctServiceMetadata [-ServiceHostId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AcctServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AcctServiceMetadata [-InputObject] <Service[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given Service.

Related topics

Set-AcctServiceMetadata

Parameters

- **ServiceHostId**<Guid>

Id of the Service

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

- **InputObject**<Service[]>

Objects to which metadata is to be removed.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue) |

- **Map**<PSObject>
Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-Name**<String>

The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

--------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AcctService | % { Remove-AcctServiceMetadata -Map $_.MetadataMap }
Remove all metadata from all Service objects.
Rename-AcctIdentityPool

Oct 10, 2013
Renames an identity pool.

Syntax


Rename-AcctIdentityPool -IdentityPoolUid <Guid> -NewIdentityPoolName <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to change the name of an existing identity pool.

Related topics

Get-AcctIdentityPool
Set-AcctIdentityPool
Remove-AcctIdentityPool
Test-AcctIdentityPoolNameAvailable

Parameters

-IdentityPoolName <String>
The name of the identity pool to be renamed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-IdentityPoolUid <Guid>
The unique identifier for the identity pool to be renamed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-NewIdentityPoolName <String>
The new name for the identity pool. This must be a name which is not used by an existing identity pool, and it must not
-PassThru

A SwitchParameter that defines whether or not the command returns a result showing the new state of the updated provisioning scheme.

Required?: false
Default Value: true
Accept Pipeline Input?: true (ByPropertyName)

LoggingId

A Guid parameter that specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Required?: false
Default Value: 
Accept Pipeline Input?: false

AdminAddress

A String parameter that specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

Required?: false
Default Value: LocalHost. Once a value is provided by any cmdlet, this value becomes the default.
Accept Pipeline Input?: false

Return Values

Citrix.AdIdentity.Sdk.IdentityPool

This object provides details of the identity pool and contains the following information:

IdentityPoolName <string>

The name of the identity pool.
IdentityPoolUid <Guid>
The unique identifier for the identity pool.

NamingScheme <string>
The naming scheme for the identity pool.

NamingSchemeType <Citrix.XDInterServiceTypes.ADIdentityNamingScheme>
The naming scheme type for the identity pool. This can be one of the following:

- Numeric - naming scheme uses numeric indexes
- Alphabetic - naming scheme uses alphabetic indexes

StartCount <int>
The next index to be used when creating an identity from the identity pool.

OU <string>
The Active Directory distinguished name for the OU in which accounts created from this identity pool will be created.

Domain <string>
The Active Directory domain that accounts in the pool belong to.

Lock <Boolean>
Indicates if the identity pool is locked.

Notes
In the case of failure the following errors can result.

Error Codes

-----------

IdentityPoolObjectNotFound
The specified identity pool could not be located.

PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error

DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-AcctIdentityPool -IdentityPoolName oldName -NewIdentityPoolName newName

Renames an existing identity pool called "oldName" to be called "newName".
Repair-AcctADAccount

Resets the AD machine password for the given accounts.

Syntax


Detailed Description

This provides the ability to synchronize the account password stored in AD with the password stored in the AD Identity Service. If successful, this results in the AD Identity Service account state being reset to 'available' so it can be consumed by other Machine Creation Services.

This command requires the user who initiated the runspace to have the required permissions in AD to reset the AD Account password.

Using this command, you can reset the password using a known existing password in both the case where AD domain administration permissions are required, and where these are NOT required.

Related topics

New-AcctADAccount
Add-AcctADAccount
Remove-AcctADAccount
Unlock-AcctADAccount
Update-AcctADAccount
Get-AcctADAccount

Parameters

-ADAccountName <String[]>

The AD account name(s) that are to be repaired. AD accounts are accepted in the following formats: Fully qualified DN e.g. CN=MyComputer,OU=Computers,DC=MyDomain,DC=Com; UPN format e.g MyComputer@MyDomain.Com; Domain qualified e.g MyDomain\MyComputer.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
### `-ADAccountSid`<String[]>

The AD account SID(s) that are to be repaired.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### `-Password`<String>

The password for the computer account.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### `-SecurePassword`<SecureString>

The password for the account (provided in a Secure String class).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### `-Force`<SwitchParameter>

Indicates whether accounts that are marked as 'in-use' can be repaired or not.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### `-LoggingId`<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

## Return Values

Citrix.AdIdentity.Sdk.AccountOperationSummary

The Repair-AcctADAccount command returns an object with the following parameters:

- **SuccessfulAccountsCount**<int>
  
The number of accounts that were repaired successfully.

- **FailedAccountsCount**<int>
  
The number of accounts that were not repaired.

- **FailedAccounts**<Citrix.AdIdentity.Sdk.AccountError[]>
  
The list of accounts that failed to be repaired. Each one has the following parameters:

  - **ADAccountName**<string>
  
  - **ADAccountSid**<String>
  
  - **ErrorReason**<AdIdentityStatus>

    This can be one of the following:

    - UnableToConvertDomain
    - IdentityNotLocatedInDomain
    - IdentityNotFound
    - IdentityObjectInUse
    - IdentityObjectLocked
    - ADServiceDatabaseError
    - ADServiceDatabaseNotConfigured
    - ADServiceStatusInvalidDb
FailedToConnectToDomainController
FailedToExecuteSearchInAD
FailedToAccessComputerAccountInAD
FailedToSetPasswordInAD
FailedToChangePasswordInAD
DiagnosticInformation <Exception>
Any other error information

Notes
In the case of failure, the following errors can result.

Error Codes

--------
PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS>Repair-AcctADAccount -ADAccountName "Domain\account","domain\account2"

<table>
<thead>
<tr>
<th>SuccessfulAccountsCount</th>
<th>FailedAccountsCount</th>
<th>FailedAccounts</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>{}</td>
</tr>
</tbody>
</table>
Reset-AcctServiceGroupMembership

Oct 10, 2013
Replaces the access permissions and service locations for the AdIdentity Service.

Syntax

Reset-AcctServiceGroupMembership [-ConfigServiceInstance] <ServiceInstance[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Enables you to reload AdIdentity Service access permissions and service locations. The Reset-AcctServiceGroupMembership command must be run on at least one instance of the service type (Acct) after installation and registration with the configuration service. Without this operation, the AdIdentity services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-AcctServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Related topics

Parameters

-ConfigServiceInstance <ServiceInstance[]>

Specifies the configuration service instance object that represents the service instance for the type 'InterService' that references a configuration service for the deployment.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-AcctServiceGroupMembership command.

Notes
If the command fails, the following errors can be returned.

Error Codes
---------

NoSuitableServiceInstance
None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-AcctServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config -AdminAddress OtherServer.example.com | Reset-AcctServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.
Set-AcctDBConnection

Oct 10, 2013
Configures a database connection for the AdIdentity Service.

Syntax
Set-AcctDBConnection [-DBConnection] <String> [-Force] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description
Configures a connection to a database in which the AdIdentity Service can store its state. The service will attempt to connect and start using the database immediately after the connection is configured. The database connection string is updated to the specified value regardless of whether it is valid or not. Specifying an invalid connection string prevents a service from functioning until the error is corrected.

After a connection is configured, you cannot alter it without first clearing it (by setting the connection to $null).

You do not need to configure a database connection to use this command.

Related topics
Get-AcctServiceStatus
Get-AcctDBConnection
Test-AcctDBConnection

Parameters
-**DBConnection**<String>
Specifies the database connection string to be used by the AdIdentity Service. Passing in $null will clear any existing database connection configured.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Force**<SwitchParameter>
If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
**AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**

```
```

The Set-AcctDBConnection command returns an object containing the status of the AdIdentity Service together with extra diagnostics information.

**DBUnconfigured**

The AdIdentity Service does not have a database connection configured.

**DBRejectedConnection**

The database rejected the logon attempt from the AdIdentity Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**

The expected stored procedures are missing from the database. This may be because the AdIdentity Service schema has not been added to the database.

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**

The version of the AdIdentity Service currently in use is incompatible with the version of the AdIdentity Service schema on the database. Upgrade the AdIdentity Service to a more recent version.

**DBOlderVersionThanService**

The version of the AdIdentity Service schema on the database is incompatible with the version of the AdIdentity Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**

A database schema upgrade is currently in progress.

**OK**

The AdIdentity Service is running and is connected to a database containing a valid schema.

**Failed**

The AdIdentity Service has failed.

**Unknown**
The status of the AdIdentity Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

InvalidDBConnectionString
The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured
There was already a database connection configured. After a configuration is set, it can only be set to $null.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

---------------------------- EXAMPLE 1 ----------------------------

c:\PS>Set-AcctDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"

OK
Configures a database connection string for the AdIdentity Service.

---------------------------- EXAMPLE 2 ----------------------------

c:\PS>Set-AcctDBConnection -DBConnection "Invalid Connection String"

Invalid database connection string format.
Configures an invalid database connection string for the AdIdentity Service.
Set-AcctIdentityPool

Oct 10, 2013
Update parameters of an identity pool.

Syntax


Detailed Description

Provides the ability to modify the parameters of an identity pool.

Note: When changing a naming scheme or naming scheme type, the index is not reset to 0; it continues to avoid AD account name clashes with existing accounts. If required, use the New-AcctAdAccount command to change the index when creating further accounts.

Related topics

New-AcctIdentityPool
Get-AcctIdentityPool
Remove-AcctIdentityPool

Parameters

-IdentityPoolName <String>

The name of the identity pool that is to be modified.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-IdentityPoolUid <Guid>

The unique identifier for the identity pool that is to be updated.

<p>| Required? | true |
| Default Value | |</p>
<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>false</th>
</tr>
</thead>
</table>

**-NamingScheme**: `<String>`

The new naming scheme that is to be used for the identity pool.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-NamingSchemeType**: `<ADIdentityNamingScheme>`

The new naming scheme type that is to be used for the identity pool. This can be Numeric or Alphabetic.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-OU**: `<String>`

The new OU to be used for the Identity Pool. All accounts created after this is set are created in this AD container. This will not move any of the existing accounts. The OU must be a valid AD container and of the domain specified for the pool.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Domain**: `<String>`

The new Active Directory domain that is to be used for the identity pool. All new accounts will be created in this domain, but this will not impact any of the existing accounts. The domain can be specified in either long or short form (i.e. domain or domain.com).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AllowUnicode**: `<SwitchParameter>`

Updates the definition of the allowed characters in a naming scheme.

| | |
| | |
- **PassThru**<SwitchParameter>

  Defines whether the command returns the new state of the identity pool or not.

- **StartCount**<Int32>

  The start index for the next create operation.

- **LoggingId**<Guid>

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

---

**Return Values**

- Required?  |  false
- Default Value |  false
- Accept Pipeline Input? |  false

**-PassThru**

Defines whether the command returns the new state of the identity pool or not.

- Required? |  false
- Default Value |  false
- Accept Pipeline Input? |  false

**-StartCount**

The start index for the next create operation.

- Required? |  false
- Default Value |  
- Accept Pipeline Input? |  false

**-LoggingId**

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

- Required? |  false
- Default Value |  
- Accept Pipeline Input? |  false

**-AdminAddress**

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

- Required? |  false
- Default Value |  LocalHost. Once a value is provided by any cmdlet, this value becomes the default.
- Accept Pipeline Input? |  false
This object provides details of the identity pool and contains the following information:

**IdentityPoolName** <string>
The name of the identity pool.

**IdentityPoolUid** <Guid>
The unique identifier for the identity pool.

**NamingScheme** <string>
The naming scheme for the identity pool.

**NamingSchemeType** <Citrix.XDInterServiceTypes.ADIdentityNamingScheme>
The naming scheme type for the identity pool. This can be one of the following:
- Numeric - naming scheme uses numeric indexes
- Alphabetic - naming scheme uses alphabetic indexes

**StartCount** <int>
The next index to be used when creating an identity from the identity pool.

**OU** <string>
The Active Directory distinguished name for the OU in which accounts created from this identity pool will be created.

**Domain** <string>
The Active Directory domain that accounts in the pool belong to.

**Lock** <Boolean>
Indicates whether the identity pool is locked.

**Notes**
In the case of failure, the following errors can result.

**Error Codes**

--------

**InvalidIdentityPoolParameterCombination**
Caused by either of the following validation errors:

* If an OU is specified then a domain must also be specified.

* NamingScheme, NamingSchemeType and Domain must all be present if any of them are specified.
NamingSchemeIllegalComputerName
The naming scheme supplied is not valid.

UnableToConvertDomainName
Unable to convert domain name to DNS format.

NamingSchemeNotEnoughCharacters
Naming scheme does not have enough characters specified.

NamingSchemeTooManyCharacters
Naming scheme has too many characters specified.

NamingSchemeIllegalCharacter
Naming scheme contains illegal characters.

NamingSchemeMayNotStartWithPeriod
Naming scheme starts with a period (.) character.

NamingSchemeMayNotBeAllNumbers
Naming scheme contains only numbers.

NamingSchemeMissingNumericSpecifications
Naming scheme does not contain any variable specification (i.e. no '#' characters are specified).

NamingSchemeHasMoreThanOneSetOfHashes
Naming scheme has more than one variable region (i.e. there are '#' characters separated by other characters).

IdentityPoolDuplicateObjectExists
An identity pool with the same name exists already.

IdentityPoolObjectNotFound
The identity pool to be modified could not be located.

IdentityPoolOUInvalid
Identity Pool OU invalid as it does not exist.

IdentityPoolOUOfWrongDomain
IdentityPool OU invalid as it refers to a different domain to the domain specified for the pool.

PermissionDenied
The user does not have administrative rights to perform this operation.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for for various reasons.

CommunicationError
An error occurred while communicating with the service.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
C:\PS>Set-AcctIdentityPool -IdentityPoolName poolName -StartCount 100 -NamingScheme AC####
Changes the start count, and the naming scheme, so that the next account generated will be AC0100 (assuming that account does not already exist)
Set-AcctIdentityPoolMetadata

Oct 10, 2013
Adds or updates metadata on the given IdentityPool.

Syntax
Set-AcctIdentityPoolMetadata [-IdentityPoolUid] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-AcctIdentityPoolMetadata [-IdentityPoolUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-AcctIdentityPoolMetadata [-IdentityPoolName] <String> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-AcctIdentityPoolMetadata [-IdentityPoolName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-AcctIdentityPoolMetadata [-InputObject] <IdentityPool[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-AcctIdentityPoolMetadata [-InputObject] <IdentityPool[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Provides the ability for additional custom data to be stored against given IdentityPool objects.

Related topics
Remove-AcctIdentityPoolMetadata

Parameters

- **-IdentityPoolUid** <Guid>
Id of the IdentityPool

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-IdentityPoolName** <String>
Name of the IdentityPool

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>
### -InputObject 
<IdentityPool[]>

Objects to which the metadata is to be added.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

### -Name <String>

Specifies the property name of the metadata to be added. The property must be unique for the IdentityPool specified. The property cannot contain any of the following characters `/;.*?<>[]()

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

### -Value <String>

Specifies the value for the property.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

### -Map <PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @("name1" = "val1"; "name2" = "val2")) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

### -LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
Accept Pipeline Input?  false

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


Set-AcctIdentityPoolMetadata returns a dictionary containing the new (name, value)-pairs.

**Key <string>**

Specifies the name of the property.

**Value <string>**

Specifies the value for the property.

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

----------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.
PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Set-AccIdentityPoolMetadata -IdentityPoolUid 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the IdentityPool with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Set-AcctServiceMetadata

Oct 10, 2013
Adds or updates metadata on the given Service.

Syntax

Set-AcctServiceMetadata [-ServiceHostId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AcctServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AcctServiceMetadata [-InputObject] <Service[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AcctServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Allows you to store additional custom data against given Service objects.

Related topics

Remove-AcctServiceMetadata

Parameters

-ServiceHostId<Guid>

Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-InputObject <Service[]>

Objects to which metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name<String>

Specifies the property name of the metadata to be added. The property must be unique for the Service specified.
property cannot contain any of the following characters \/:.*?<>|[]()“

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

- **Value**<String>

Specifies the value for the property.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

- **Map**<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

- **LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
Accept Pipeline Input? | false

Return Values
Set-AcctServiceMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>
Specifies the name of the property.

Value <string>
Specifies the value for the property.

Notes
If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

**CommunicationError**

There was a problem communicating with the remote service.

**ExceptionThrown**

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

**Examples**

------------------------------- EXAMPLE 1 ------------------------------

c:\PS>Set-AcctServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Test-AcctDBConnection

Oct 10, 2013
Tests a database connection for the AdIdentity Service.

Syntax

Test-AcctDBConnection [-DBConnection] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Tests a connection to the database in which the AdIdentity Service can store its state. The service will attempt to connect to the database without affecting the current connection to the database.

You do not have to clear the connection to use this command.

Related topics

Get-AcctServiceStatus

Get-AcctDBConnection

Set-AcctDBConnection

Parameters

-DBConnection <String>

Specifies the database connection string to be tested by the AdIdentity Service.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

https://docs.citrix.com
The Test-AcctDBConnection command returns an object containing the status of the AdIdentity Service if the connection string of the specified data store were to be set to the string being tested, together with extra diagnostics information for the specified connection string.

**DBRejectedConnection**

The database rejected the logon attempt from the AdIdentity Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**

The expected stored procedures are missing from the database. This may be because the AdIdentity Service schema has not been added to the database.

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**

The version of the AdIdentity Service currently in use is incompatible with the version of the AdIdentity Service schema on the database. Upgrade the AdIdentity Service to a more recent version.

**DBOlderVersionThanService**

The version of the AdIdentity Service schema on the database is incompatible with the version of the AdIdentity Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**

A database schema upgrade is currently in progress.

**OK**

The Set-AcctDBConnection command would succeed if it were executed with the supplied connection string.

**Failed**

The AdIdentity Service has failed.

**Unknown**

The status of the AdIdentity Service cannot be determined.

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

--------

**InvalidDBConnectionString**

The database connection string has an invalid format.

**PermissionDenied**

You do not have permission to execute this command.

**AuthorizationError**

There was a problem communicating with the Citrix Delegated Administration Service.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Test-AcctDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Tests a database connection string for the AdIdentity Service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Test-AcctDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Tests an invalid database connection string for the AdIdentity Service.
Test-AcctIdentityPoolNameAvailable

Oct 10, 2013
Checks to ensure that the proposed name for an identity pool is unused.

Syntax

Test-AcctIdentityPoolNameAvailable [-IdentityPoolName] <String[]> [-AdminAddress <String>] [CommonParameters]

Detailed Description

Checks to ensure that the proposed name for an identity pool is unused. This check is done without regard for scoping of existing identity pools, so the names of inaccessible pools are also checked.

Related topics

New-AcctIdentityPool
Rename-AcctIdentityPool

Parameters

-IdentityPoolName <String[]>
The name or names of the identity pool(s) to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

object[]
An array of PSObjects that pair the name and availability of the name

Notes

In the case of failure, the following errors can result.
Error Codes

-----------

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

Test-AcctIdentityPoolNameAvailable -IdentityPoolName $NewPoolName
This tests whether the value of $NewPoolName is unique or not, and can be used to create a new provisioning scheme or rename an existing one without failing. True is returned if the name is good.
Unlock-AcctADAccount

Oct 10, 2013
Unlocks AD accounts within the AD Identity Service.

Syntax

Unlock-AcctADAccount -ADAccountName <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Unlock-AcctADAccount -ADAccountSid <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to unlock the AD Identity Service identity item that references a specified AD account. An AD account is marked as locked in the AD Identity Service while the Machine Creation Services (MCS) are processing tasks relating to the account. If these tasks are forcibly stopped, an account can remain locked despite no longer being processed. This command resolves this issue, but use it with caution because unlocking an account that MCS expects to be locked can result in an MCS operation being cancelled. Use this command only when MCS has locked an account for use in a provisioning operation, and where this operation has failed without unlocking the account.

Note: This command does NOT make any changes to the account information stored in Active Directory.

Related topics

Get-AcctADAccount
New-AcctADAccount
Add-AcctADAccount
Repair-AcctADAccount
Remove-AcctADAccount
Update-AcctADAccount
Unlock-AcctADAccount

Parameters

-ADAccountName <String>

The AD account name to be unlocked. AD account name is accepted in the following formats: Fully qualified DN e.g. CN=MyComputer,OU=Computers,DC=MyDomain,DC=Com; UPN format e.g MyComputer@MyDomain.Com; Domain qualified e.g MyDomain\MyComputer.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
**-ADAccountSid**: <String>

The AD account SID that represents the account to be unlocked.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-LoggingId**: <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**: <String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.AdIdentity.Sdk.IdentityInPool

You can pipe an object containing a parameter called 'ADAccountSID' to unlock-AcctADAccount.

**Notes**

In the case of failure, the following errors can result.

**Error Codes**

--------

IdentityNotLocatedInDomain

The specified AD account could not be located in Active Directory.

IdentityObjectNotFound
The identity could not be found.
IdentityAlreadyUnlocked
The identity is not locked.
PermissionDenied
The user does not have administrative rights to perform this operation.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error
DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.
CommunicationError
An error occurred while communicating with the service.
ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> Unlock-AcctADAccount -ADAccountName Domain\account
Unlocks the AD account called "Domain\account".

-------------------------- EXAMPLE 2 --------------------------

c:\PS> Get-AcctADAccount -Filter {lock -eq $true} | Unlock-AcctADAccount
Unlocks all the locked AD accounts.
Unlock-AcctIdentityPool

Oct 10, 2013
Unlocks identity pools.

Syntax

Unlock-AcctIdentityPool [-IdentityPoolName] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Unlock-AcctIdentityPool -IdentityPoolUid <Guid> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to unlock the specified identity pool. Identity pools are locked automatically when being updated (e.g. when new accounts are being created into them). The pool must never be left in a locked state; this command allows recovery from an error should this ever occur. Use this command with caution, as unlocking an identity pool which is supposed to be locked may result in unexpected behavior.

Related topics

New-AcctIdentityPool
Remove-AcctIdentityPool
Set-AcctIdentityPool

Parameters

-IdentityPoolName <String>
The name of the identity pool to unlock.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-IdentityPoolUid <Guid>
The unique identifier for the identity pool to be unlocked.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create
high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a hostname or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.Identity.Sdk.IdentityPool You can pipe an object containing a parameter called 'IdentityPoolName' to unlock-AcctIdentityPool.

**Notes**

In the case of failure, the following errors can result.

**Error Codes**

------------

IdentityPoolObjectNotFound

The specified identity pool could not be located.

IdentityPoolAlreadyUnlocked

The specified identity pool is not locked.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

DatabaseError

An error occurred in the service while attempting a database operation.
DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Unlock-AcctIdentityPool -IdentityPool MyPool
Unlocks the identity pool called "MyPool".

-------------------------- EXAMPLE 2 --------------------------

C:\PS>Get-AcctIdentityPool -Filter {Lock -eq $true} | Unlock-AcctIdentityPool
Unlocks all the locked identity pools.
Update-AcctADAccount

Oct 10, 2013
Refreshes the AD computer account state stored in the AD Identity Service.

Syntax

Update-AcctADAccount [-IdentityPoolName <String>] [-AllAccounts] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Update-AcctADAccount -IdentityPoolUid <Guid> [-AllAccounts] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to synchronize the state of the AD accounts stored in the AD Identity Service with the AD accounts themselves. By default, this checks all accounts marked as 'error' to determine if accounts are still in an error state (i.e. disabled or locked). If you specify the 'AllAccounts' option, it checks accounts not in error state and updates the status of these accounts too.

Related topics

New-AcctADAccount
Add-AcctADAccount
Remove-AcctADAccount
Repair-AcctADAccount
Unlock-AcctADAccount
Get-AcctADAccount

Parameters

-IdentityPoolName <String>

The name of the identity pool for the accounts to be refreshed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-IdentityPoolUid <Guid>

The unique identifier for the identity pool of the AD accounts that are to be refreshed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
### Default Value

| Accept Pipeline Input? | false |

---

### -AllAccounts

<SwitchParameter>

Indicates if all accounts should be refreshed or only the ones marked as in error.

| Required? | false |
| Default Value | false |
| Accept Pipeline Input? | false |

---

### -LoggingId

<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

---

### -AdminAddress

<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | LocalHost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

---

### Notes

In the case of failure, the following errors can result.

#### Error Codes

----------

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS> Update-AcctADAccount -IdentityPoolName MyPool
Checks the status of all accounts in the Identity Pool MyPool that are currently mark as in the Error state and checks if the account is now available.

-------------------------- EXAMPLE 2 --------------------------
c:\PS> Update-AcctADAccount -IdentityPoolName MyPool -AllAccounts
Checks the status of all accounts in the Identity Pool MyPool marking them as Available, InUse, Tainted or Error as appropriate.
### Cmdlets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvertTo-CtxAppVLauncherArg</td>
<td>Returns a string containing information to send to the App-V Launcher. You can plug this string directly into the Virtual Delivery Agent (VDA) to launch App-V applications.</td>
</tr>
<tr>
<td>Get-CtxAppVApplication</td>
<td>Enumerates all published App-V applications for a given Management server.</td>
</tr>
<tr>
<td>Get-CtxAppVApplicationInfo</td>
<td>Enumerates information for a given application in a given package for a given Management server.</td>
</tr>
<tr>
<td>Get-CtxAppVServer</td>
<td>Returns URLs for App-V Publishing and Management servers contained in a Citrix App-V policy. Returned values are in string format.</td>
</tr>
<tr>
<td>Set-CtxAppVServerSetting</td>
<td>Specifies the App-V Publishing server settings to use on the VDA. These settings determine whether or not the App-V Client can automatically initiate a publishing refresh on certain events such as user logon or at specified intervals.</td>
</tr>
<tr>
<td>Test-CtxAppVServer</td>
<td>Tests the given URL for the presence of App-V Management and Publishing servers.</td>
</tr>
</tbody>
</table>

---

**Citrix.AppV.Admin.V1**

Oct 10, 2013

**Cmdlets**
ConvertTo-CtxAppVLauncherArg

Oct 10, 2013
Returns a string containing information to send to the App-V Launcher. You can plug this string directly into the Virtual Delivery Agent (VDA) to launch App-V applications.

Syntax
ConvertTo-CtxAppVLauncherArg [-LauncherPath] <SwitchParameter> [CommonParameters]

Detailed Description
Returns a string containing information to send to the App-V Launcher. You can plug this string directly into the Virtual Delivery Agent (VDA) to launch App-V applications.

Related topics
Parameters
-AppVPublishingServer
  Required? true
  Default Value
  Accept Pipeline Input? true (ByValue, ByPropertyName)

-PackageId
  The Package Id of the application.
  Required? false
  Default Value
  Accept Pipeline Input? true (ByValue, ByPropertyName)

-AppId
  The App Id of the application.
  Required? True
  Default Value
  Accept Pipeline Input? false

-SeqLoc
  The sequence location of the package.
  Required? true
  Default Value
  Accept Pipeline Input? false

-TargetInPackage
  Pass this as $true if TargetInPackage field is "true" in the Manifest file for the particular App Id.
  Required? true
  Default Value
  Accept Pipeline Input? false

-LauncherPath
  Gets the Citrix Launcher path. The Citrix Launcher is a component installed on the VDA.
  Required? true
  Default Value
  Accept Pipeline Input? true (ByValue, ByPropertyName)

Examples
-------------------------- EXAMPLE 1 --------------------------
ConvertTo-CtxAppVLauncherArg -AppVPublishingServer "http://appv-2k82-srv.blrstrm.com:8082" -PackageId "1bcb6993-1d1-4659-b0dd-e009e10cecd/5" -AppName "|||ProgramFilesX86|||Beyond Compare"
Converts the arguments provided into a cmdlet to launch Beyond Compare from http://appv-server on the VDA.

---------------------------- EXAMPLE 2 -----------------------------

ConvertTo-CtxAppVLauncherArg -LauncherPath
Gets the Citrix App-V Launcher path.
Get-CtxAppVApplication

Oct 10, 2013
Enumerates all published App-V applications for a given Management server.

Syntax
Get-CtxAppVApplication [-AppVMangementServer] <string> [<CommonParameters>]

Detailed Description
Queries a given Management server and fetches all published applications for that server. Applications that are published but have no user Entitlements are not displayed by this cmdlet.

Related topics
Parameters

-AppVMangementServer <String>

App-V Management Server

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

Examples

-------------------------- EXAMPLE 1 --------------------------
Get-CtxAppVApplication -AppVMangementServer "xmas-demo-appv"
Displays all published applications for the Management server "xmas-demo-appv".
Get-CtxAppVApplicationInfo

Oct 10, 2013
Enumerates information for a given application in a given package for a given Management server.

Syntax

Detailed Description
Queries a given Management server and fetches the requested information for a given application.

Related topics
Parameters
-AppvManagementServer <string>
Machine name of the App-V Management server. The name does not need to be specified as a Fully Qualified Domain Name (FQDN).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

Accept Pipeline Input? true [ByValue]

-AppId <string>
The App Id of the given application.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

-PackageId <string>
The Package Id of the given application.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

Return Values

Examples
**************************** EXAMPLE 1 ****************************
Get-CtxAppVApplicationInfo -AppVManagementServer "xmas-demo-appv" -AppId "\{ProgramFilesX86\}\Beyond Compare 3\BCompare.exe" -PackageId "1bcb6993-10b1-4659-b9d4-e809e10cecdf_5" Fetches all application information for the Beyond Compare 3 application from the xmas-demo-appv Management server.

**************************** EXAMPLE 2 ****************************
Get-CtxAppVApplicationInfo -AppVManagementServer "xmas-demo-appv" -AppId "\{ProgramFilesX86\}\Beyond Compare 3\BCompare.exe" -PackageId "1bcb6993-10b1-4659-b9d4-e809e10cecdf_5" -I Fetches only information about FTA and User Entitlements for the Beyond Compare 3 application from the xmas-demo-appv Management server.

**************************** EXAMPLE 3 ****************************
Get-CtxAppVApplicationInfo -AppVManagementServer "xmas-demo-appv" | Get-CtxAppVApplicationInfo -AppVManagementServer "xmas-demo-appv" Pipelines the output of Get-CtxAppVApplicationInfo to get the application properties of all the published applications on the xmas-demo-appv Management server.
Get-CtxAppVServer

Oct 10, 2013

Returns URLs for App-V Publishing and Management servers contained in a Citrix App-V policy. Returned values are in string format.

Syntax

Get-CtxAppVServer -ByteArray <byte[]> [-ConsumedByStudio <bool>] [CommonParameters]

Detailed Description

Returns URLs for App-V Publishing and Management servers contained in a Citrix App-V policy. Returned values are in string format.

Related topics

Parameters

-ByteArray

Specifies the Citrix App-V policy created using the New-CtxAppVServer cmdlet.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-ConsumedByStudio

If set to "true", outputs URLs for the App-V Publishing and Management servers. If set to "false", outputs both the URL and settings for the App-V Publishing server.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

Examples

-------------------------- EXAMPLE 1 --------------------------

$config = Get-BrokerMachineConfiguration -Name appv*
Get-CtxAppVServer -ByteArray $config[0].Policy

Returns Publishing Server URL, Management Server URL configured in given policy
Get-CtxAppVServerSetting

Oct 10, 2013
Returns settings for the specified App-V Publishing Server.

Syntax
Get-CtxAppVServerSetting -AppVPublishingServer <string> [CommonParameters]

Detailed Description

Related topics
Parameters
-AppVPublishingServer
URL of the Publishing Server for which settings are to be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

Return Values
Publishing Server settings. These settings are: GlobalRefreshEnabled; GlobalRefreshOnLogon;GlobalRefreshInterval;GlobalRefreshIntervalUnit;UserRefreshEnabled; UserRefreshOnLogon;UserRefreshInterval;UserRefreshIntervalUnit;

Examples
-------------------------- EXAMPLE 1 --------------------------
This example returns settings associated with http://AppV50PublishingServer:8082 in the following format:

GlobalRefreshEnabled: false
GlobalRefreshOnLogon: false
GlobalRefreshInterval: 0
GlobalRefreshIntervalUnit: Day
UserRefreshEnabled: true
UserRefreshOnLogon: false
UserRefreshInterval: 0

UserRefreshIntervalUnit: Hour
New-CtxAppVServer

Oct 10, 2013

Creates a new Citrix App-V policy containing the specified App-V Management and Publishing server URLs.

Syntax


Detailed Description

Creates a new Citrix App-V policy containing the specified App-V Management and Publishing server URLs. Additionally, accepts Publishing Server settings that control how and when automatic refresh occurs on the VDA.

Related topics

Parameters

-PublishingServer

URL of the Publishing server to add to the Citrix policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-ManagementServer

URL of the Management server to add to the Citrix policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-UserRefreshEnabled

Enables a refresh of packages published to user groups either at user logon or at a specified interval. For more information, see the App-V 5.0 documentation at http://technet.microsoft.com/en-us/library/jj687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-UserRefreshOnLogon

Specifies whether or not to initiate a refresh of packages published to user groups on every user logon. For more information, see the App-V 5.0 documentation at http://technet.microsoft.com/en-us/library/jj687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-UserRefreshInterval

Specifies the frequency at which to initiate a refresh of packages published to user groups. This can be either days or hours, as specified by the UserRefreshIntervalUnit setting. For more information, see the App-V 5.0 documentation at http://technet.microsoft.com/en-us/library/jj687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-UserRefreshInterval

Specifies the frequency at which to initiate a refresh of packages published to user groups. This can be either days or hours, as specified by the UserRefreshIntervalUnit setting. For more information, see the App-V 5.0 documentation at http://technet.microsoft.com/en-us/library/jj687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.628
### Accept Pipeline Input?

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>false</th>
</tr>
</thead>
</table>

**-UserRefreshIntervalUnit**

Specifies the unit for the UserRefreshInterval setting. This can be set to either Hours (0) or Days (1). For more information, see the App-V 5.0 documentation at http://technet.microsoft.com/en-us/library/j687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### Required?

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### GlobalRefreshEnabled

Enables a refresh of packages published to machine groups either at user logon or at a specified interval. For more information, see the App-V 5.0 documentation at http://technet.microsoft.com/en-us/library/j687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>True</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### GlobalRefreshOnLogon

Specifies whether or not to initiate a refresh of packages published to machine groups on every user logon. For more information, see the App-V 5.0 documentation at http://technet.microsoft.com/en-us/library/j687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### GlobalRefreshInterval

Specifies the frequency at which to initiate a refresh of packages published to machine groups. This can be either days or hours, as specified by the GlobalRefreshIntervalUnit setting. Please refer to App-V 5.0 documentation for details. http://technet.microsoft.com/en-us/library/j687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### GlobalRefreshIntervalUnit

Specifies the unit for the GlobalRefreshInterval setting. This can be set to either Hours (0) or Days (1). Please refer to App-V 5.0 documentation for details. http://technet.microsoft.com/en-us/library/j687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### Examples

**-------------------------- EXAMPLE 1 --------------------------**


Default values for publishing server settings used for Http://AppV-Mgmt-Server:8082 are:

- GlobalRefreshEnabled = false; GlobalLogonRefresh = false; GlobalIntervalRefreshInterval = 0; GlobalIntervalRefreshUnit = Day
- UserRefreshEnabled = true; UserLogonRefresh = true; UserIntervalRefreshInterval = 0; GlobalIntervalRefreshUnit = Day

**-------------------------- EXAMPLE 1 --------------------------**


Following values are used to configure Publishing Server: http://AppV-Mgmt-Server:8082

- GlobalRefreshEnabled = True; GlobalLogonRefresh = True; GlobalIntervalRefreshInterval = 0; GlobalIntervalRefreshUnit = Day
- UserRefreshEnabled = true; UserLogonRefresh = true; UserIntervalRefreshInterval = 0; GlobalIntervalRefreshUnit = Day
Set-CtxAppVServerSetting

Oct 10, 2013

Specifies the App-V Publishing server settings to use on the VDA. These settings determine whether or not the App-V Client can automatically initiate a publishing refresh on certain events such as user logon or at specified intervals.

Syntax


Detailed Description

Specifies the App-V Publishing server settings to use on the VDA. These settings determine whether or not the App-V Client can automatically initiate a publishing refresh on certain events such as user logon or at specified intervals. Please refer to Microsoft App-V 5.0 documentation for more details on these settings: http://technet.microsoft.com/en-us/library/jj687745.aspx

Related topics

Parameters

-AppVPublishingServer

URL of the Publishing Server for which settings are to be set.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

UserRefreshEnabled

Enables a refresh of packages published to user groups either at user logon or at a specified interval. For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>True</td>
<td>false</td>
</tr>
</tbody>
</table>

UserRefreshOnLogon

Specifies whether or not to initiate a refresh of packages published to user groups on every user logon. For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

UserRefreshInterval

Specifies the frequency at which to initiate a refresh of packages published to user groups. This can be either days or hours, as specified by the UserRefreshIntervalUnit setting. For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

UserRefreshIntervalUnit

Specifies the unit for the UserRefreshInterval setting. This can be set to either Hours (0) or Days (1). For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

GlobalRefreshEnabled

Enables a refresh of packages published to machine groups either at user logon or at a specified interval. For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>True</td>
<td>false</td>
</tr>
</tbody>
</table>
-GlobalRefreshOnLogon

Specifies whether or not to initiate a refresh of packages published to machine groups on every user logon. For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

Required? true
Default Value false
Accept Pipeline Input? false

-GlobalRefreshInterval

Specifies the frequency at which to initiate a refresh of packages published to machine groups. This can be either days or hours, as specified by the GlobalRefreshIntervalUnit setting. For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

Required? true
Default Value
Accept Pipeline Input? false

-GlobalRefreshIntervalUnit

Specifies the unit for the GlobalRefreshInterval setting. This can be set to either Hours (0) or Days (1). For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

Required? true
Default Value
Accept Pipeline Input? false

Examples

---------------------------- EXAMPLE 1 --------------------------
Test-CtxAppVServer

Oct 10, 2013
Tests the given URL for the presence of App-V Management and Publishing servers.

Syntax

 Test-CtxAppVServer [-AppVManagementServer] <string> [CommonParameters]
 Test-CtxAppVServer [-AppVPublishingServer] <string> [CommonParameters]

Detailed Description
Tests the given URL for the presence of App-V Management and Publishing servers.

Related topics

Parameters

- **-AppVManagementServer**: String

  Machine name of the App-V Management server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-AppVPublishingServer**: String

  Machine name of the App-V Publishing server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Microsoft.AppvAgent.AppvClientPackage

Examples

-------------------------- EXAMPLE 1 --------------------------
Test-CtxAppVServer -AppVManagementServer "xmas-demo-appv"
Tests whether "xmas-demo-appv" is a Management server or not. The name does not need to be specified as a Fully Qualified Domain Name (FQDN).

-------------------------- EXAMPLE 2 --------------------------
Tests whether "http://appvb2refserver.blrstrm.com:8082" is a Publishing server or not. Specify the full address, including FQDN and port number, of the Publishing server.
## Citrix.Broker.Admin.V2

### Oct 10, 2013

### Overview

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broker AccessPolicy</td>
<td>Controls client-connection-based access to desktop groups.</td>
</tr>
<tr>
<td>Broker Applications</td>
<td>Describes how to publish and manage hosted applications.</td>
</tr>
<tr>
<td>Broker AssignmentPolicy</td>
<td>Controls the automatic, permanent assignment of machines to users.</td>
</tr>
<tr>
<td>Broker Concepts</td>
<td>Overview of the Citrix Broker.</td>
</tr>
<tr>
<td>Broker ConfigurationSlots</td>
<td>Overview of assigning a collection of related settings to a desktop group.</td>
</tr>
<tr>
<td>Broker ControllerDiscovery</td>
<td>Describes the way that machines providing published resources discover</td>
</tr>
<tr>
<td>Broker Desktops</td>
<td>Describes desktop concepts and usage.</td>
</tr>
<tr>
<td>Broker EntitlementPolicy</td>
<td>Controls end-user entitlement to desktop and application sessions provided</td>
</tr>
<tr>
<td>Broker ErrorHandling</td>
<td>Describes broker errors generated by cmdlets and how to access them.</td>
</tr>
<tr>
<td>Broker Filtering</td>
<td>Describes the common filtering options for XenDesktop cmdlets.</td>
</tr>
<tr>
<td>Broker Licensing</td>
<td>Overview of broker licensing configuration.</td>
</tr>
<tr>
<td>Broker Machines</td>
<td>Describes machine concepts and usage.</td>
</tr>
<tr>
<td>Broker Policies</td>
<td>Overview of the site policies that control users' access to desktop and</td>
</tr>
<tr>
<td>Broker PostInstallPreConfiguration</td>
<td>Describes how to configure the Citrix Broker Service port numbers, URL</td>
</tr>
<tr>
<td>Broker PowerManagement</td>
<td>Describes power management of machines used for desktops and applications.</td>
</tr>
<tr>
<td>Broker RemotePC</td>
<td>Overview of the Remote PC feature.</td>
</tr>
</tbody>
</table>

### Cmdlets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-BrokerApplication</td>
<td>Adds applications to a desktop group.</td>
</tr>
<tr>
<td>Add-BrokerDesktopGroup</td>
<td>Associate Remote PC desktop groups with the specified Remote PC catalog.</td>
</tr>
<tr>
<td>Add-BrokerMachine</td>
<td>Adds one or more machines to a desktop group.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Add-BrokerMachineConfiguration</td>
<td>Add machine configuration to a desktop group.</td>
</tr>
<tr>
<td>Add-BrokerMachinesToDesktopGroup</td>
<td>Adds machines from a catalog to a desktop group.</td>
</tr>
<tr>
<td>Add-BrokerScope</td>
<td>Add the specified catalog/desktop group to the given scope(s).</td>
</tr>
<tr>
<td>Add-BrokerTag</td>
<td>Associate a tag with another object.</td>
</tr>
<tr>
<td>Add-BrokerUser</td>
<td>Creates an association between a user and another broker object</td>
</tr>
<tr>
<td>Disconnect-BrokerSession</td>
<td>Disconnect a session.</td>
</tr>
<tr>
<td>Export-BrokerDesktopPolicy</td>
<td>Gets the site wide Citrix Group Policy settings.</td>
</tr>
<tr>
<td>Get-BrokerAccessPolicyRule</td>
<td>Gets rules from the site's access policy.</td>
</tr>
<tr>
<td>Get-BrokerAppAssignmentPolicyRule</td>
<td>Gets application rules from the site's assignment policy.</td>
</tr>
<tr>
<td>Get-BrokerAppEntitlementPolicyRule</td>
<td>Gets application rules from the site's entitlement policy.</td>
</tr>
<tr>
<td>Get-BrokerApplication</td>
<td>Get the applications published on this site.</td>
</tr>
<tr>
<td>Get-BrokerApplicationInstance</td>
<td>Gets the running applications on the desktops.</td>
</tr>
<tr>
<td>Get-BrokerAssignmentPolicyRule</td>
<td>Gets desktop rules from the site's assignment policy.</td>
</tr>
<tr>
<td>Get-BrokerCatalog</td>
<td>Gets catalogs configured for this site.</td>
</tr>
<tr>
<td>Get-BrokerConfigurationSlot</td>
<td>Gets configuration slots configured for this site.</td>
</tr>
<tr>
<td>Get-BrokerConfiguredFTA</td>
<td>Gets any file type associations configured for an application.</td>
</tr>
<tr>
<td>Get-BrokerConnectionLog</td>
<td>Get entries from the site's session connection log.</td>
</tr>
<tr>
<td>Get-BrokerController</td>
<td>Gets Controllers running broker services in the site.</td>
</tr>
<tr>
<td>Get-BrokerDBConnection</td>
<td>Gets the Citrix Broker Service's database connection string.</td>
</tr>
<tr>
<td>Get-BrokerDBSchema</td>
<td>Gets SQL scripts to create or maintain the database schema for the Citrix Broker Service.</td>
</tr>
<tr>
<td>Get-BrokerDBVersionChangeScript</td>
<td>Gets an SQL service schema update script for the Citrix Broker Service.</td>
</tr>
<tr>
<td>Get-BrokerDelayedHostingPowerAction</td>
<td>Gets power actions that are executed after a delay.</td>
</tr>
<tr>
<td>Get-BrokerDesktop</td>
<td>Gets desktops configured for this site.</td>
</tr>
<tr>
<td>Get-BrokerDesktopGroup</td>
<td>Gets broker desktop groups configured for this site.</td>
</tr>
<tr>
<td>Get-BrokerDesktopUsage</td>
<td>Get usage history of desktop groups.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Get-BrokerEntitlementPolicyRule</td>
<td>Gets rules from the site's entitlement policy.</td>
</tr>
<tr>
<td>Get-BrokerHostingPowerAction</td>
<td>Gets power actions queued for machines.</td>
</tr>
<tr>
<td>Get-BrokerHypervisorAlert</td>
<td>Gets hypervisor alerts recorded by the controller.</td>
</tr>
<tr>
<td>Get-BrokerHypervisorConnection</td>
<td>Gets hypervisor connections matching the specified criteria.</td>
</tr>
<tr>
<td>Get-BrokerIcon</td>
<td>Get stored icons.</td>
</tr>
<tr>
<td>Get-BrokerImportedFTA</td>
<td>Gets the imported file type associations.</td>
</tr>
<tr>
<td>Get-BrokerMachine</td>
<td>Gets machines belonging to this site.</td>
</tr>
<tr>
<td>Get-BrokerMachineCommand</td>
<td>Get the list of commands queued for delivery to a desktop.</td>
</tr>
<tr>
<td>Get-BrokerMachineConfiguration</td>
<td>Gets machine configurations defined for this site.</td>
</tr>
<tr>
<td>Get-BrokerMachineStartMenuShortcutIcon</td>
<td>Retrieves a Start Menu Shortcut icon from the specified machine.</td>
</tr>
<tr>
<td>Get-BrokerMachineStartMenuShortcuts</td>
<td>Retrieves the Start Menu Shortcuts from the specified machine.</td>
</tr>
<tr>
<td>Get-BrokerPowerTimeScheme</td>
<td>Gets power management time schemes for desktop groups.</td>
</tr>
<tr>
<td>Get-BrokerPrivateDesktop</td>
<td>Get private desktops configured for this site.</td>
</tr>
<tr>
<td>Get-BrokerRebootCycle</td>
<td>Gets one or more reboot cycles.</td>
</tr>
<tr>
<td>Get-BrokerRebootSchedule</td>
<td>Gets one or more reboot schedules.</td>
</tr>
<tr>
<td>Get-BrokerRemotePCAccount</td>
<td>Get RemotePCAccount entries configured for this site.</td>
</tr>
<tr>
<td>Get-BrokerResource</td>
<td>Gets resources that a user can broker connections to.</td>
</tr>
<tr>
<td>Get-BrokerScopedObject</td>
<td>Gets the details of the scoped objects for the Broker Service.</td>
</tr>
<tr>
<td>Get-BrokerServiceAddedCapability</td>
<td>Gets any added capabilities for the Broker Service on the controller.</td>
</tr>
<tr>
<td>Get-BrokerServiceInstance</td>
<td>Gets the service instance entries for the Broker Service.</td>
</tr>
<tr>
<td>Get-BrokerServiceStatus</td>
<td>Determines the current state of the Broker Service on the controller.</td>
</tr>
<tr>
<td>Get-BrokerSession</td>
<td>Gets a list of sessions.</td>
</tr>
<tr>
<td>Get-BrokerSharedDesktop</td>
<td>Get shared desktops configured for this site.</td>
</tr>
<tr>
<td>Get-BrokerSite</td>
<td>Gets the current XenDesktop broker site.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Get-BrokerTag</td>
<td>Gets one or more tags.</td>
</tr>
<tr>
<td>Get-BrokerUnconfiguredMachine</td>
<td>Gets machines that have registered but are not yet configured in this site.</td>
</tr>
<tr>
<td>Get-BrokerUser</td>
<td>Gets broker users configured for this site.</td>
</tr>
<tr>
<td>Group-BrokerDesktop</td>
<td>Groups and counts desktops with the same value for a specified property.</td>
</tr>
<tr>
<td>Group-BrokerMachine</td>
<td>Groups and counts machines with the same value for a specified property.</td>
</tr>
<tr>
<td>Import-BrokerDesktopPolicy</td>
<td>Sets the site wide Citrix Group Policy settings for the site.</td>
</tr>
<tr>
<td>New-BrokerAccessPolicyRule</td>
<td>Creates a new rule in the site's access policy.</td>
</tr>
<tr>
<td>New-BrokerAppAssignmentPolicyRule</td>
<td>Creates a new application rule in the site's assignment policy.</td>
</tr>
<tr>
<td>New-BrokerAppEntitlementPolicyRule</td>
<td>Creates a new application rule in the site's entitlement policy.</td>
</tr>
<tr>
<td>New-BrokerApplication</td>
<td>Creates a new published application.</td>
</tr>
<tr>
<td>New-BrokerAssignmentPolicyRule</td>
<td>Creates a new desktop rule in the site's assignment policy.</td>
</tr>
<tr>
<td>New-BrokerCatalog</td>
<td>Adds a new catalog to the site.</td>
</tr>
<tr>
<td>New-BrokerConfigurationSlot</td>
<td>Creates a new configuration slot.</td>
</tr>
<tr>
<td>New-BrokerConfiguredFTA</td>
<td>Creates a file type association with a published application.</td>
</tr>
<tr>
<td>New-BrokerDelayedHostingPowerAction</td>
<td>Causes a power action to be queued after a delay.</td>
</tr>
<tr>
<td>New-BrokerDesktopGroup</td>
<td>Create a new desktop group for managing the brokering of groups of desktops.</td>
</tr>
<tr>
<td>New-BrokerEntitlementPolicyRule</td>
<td>Creates a new desktop rule in the site's entitlement policy.</td>
</tr>
<tr>
<td>New-BrokerHostingPowerAction</td>
<td>Creates a new action in the power action queue.</td>
</tr>
<tr>
<td>New-BrokerHypervisorConnection</td>
<td>Creates a new hypervisor connection.</td>
</tr>
<tr>
<td>New-BrokerIcon</td>
<td>Creates a new icon.</td>
</tr>
<tr>
<td>New-BrokerMachine</td>
<td>Adds a machine that can be used to run desktops and applications.</td>
</tr>
<tr>
<td>New-BrokerMachineCommand</td>
<td>Creates a new command to deliver to a desktop.</td>
</tr>
<tr>
<td>New-BrokerMachineConfiguration</td>
<td>Creates a new machine configuration associated with an existing configuration slot.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>New-BrokerPowerTimeScheme</td>
<td>Creates a new power time scheme for a desktop group.</td>
</tr>
<tr>
<td>New-BrokerRebootSchedule</td>
<td>Creates a new reboot schedule for a desktop group.</td>
</tr>
<tr>
<td>New-BrokerRemotePCAccount</td>
<td>Create a new RemotePCAccount.</td>
</tr>
<tr>
<td>New-BrokerTag</td>
<td>Creates a new tag.</td>
</tr>
<tr>
<td>New-BrokerUser</td>
<td>Creates a new broker user object</td>
</tr>
<tr>
<td>Remove-BrokerAccessPolicyRule</td>
<td>Deletes a rule from the site's access policy.</td>
</tr>
<tr>
<td>Remove-BrokerAppAssignmentPolicyRule</td>
<td>Deletes an application rule from the site's assignment policy.</td>
</tr>
<tr>
<td>Remove-BrokerApplication</td>
<td>Deletes one or more applications, or an association of an application.</td>
</tr>
<tr>
<td>Remove-BrokerApplicationInstanceMetadata</td>
<td>Deletes ApplicationInstance Metadata from the ApplicationInstance objects</td>
</tr>
<tr>
<td>Remove-BrokerApplicationMetadata</td>
<td>Deletes Application Metadata from the Application objects</td>
</tr>
<tr>
<td>Remove-BrokerAssignmentPolicyRule</td>
<td>Deletes a desktop rule from the site's assignment policy.</td>
</tr>
<tr>
<td>Remove-BrokerCatalog</td>
<td>Removes catalogs from the site.</td>
</tr>
<tr>
<td>Remove-BrokerCatalogMetadata</td>
<td>Deletes Catalog Metadata from the Catalog objects</td>
</tr>
<tr>
<td>Remove-BrokerConfigurationSlot</td>
<td>Removes a configuration slot.</td>
</tr>
<tr>
<td>Remove-BrokerConfigurationSlotMetadata</td>
<td>Deletes ConfigurationSlot Metadata from the ConfigurationSlot objects</td>
</tr>
<tr>
<td>Remove-BrokerConfiguredFTA</td>
<td>Deletes one or more configured file type associations.</td>
</tr>
<tr>
<td>Remove-BrokerControllerMetadata</td>
<td>Deletes Controller Metadata from the Controller objects</td>
</tr>
<tr>
<td>Remove-BrokerDelayedHostingPowerAction</td>
<td>Cancels one or more delayed power actions.</td>
</tr>
<tr>
<td>Remove-BrokerDesktopGroup</td>
<td>Remove desktop groups from the system or remove them from a Remote PC catalog.</td>
</tr>
<tr>
<td>Remove-BrokerDesktopGroupMetadata</td>
<td>Deletes DesktopGroup Metadata from the DesktopGroup objects</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Remove-BrokerEntitlementPolicyRule</td>
<td>Deletes a desktop rule from the site's entitlement policy.</td>
</tr>
<tr>
<td>Remove-BrokerHostingPowerAction</td>
<td>Cancel one or more pending power actions.</td>
</tr>
<tr>
<td>Remove-BrokerHostingPowerActionMetadata</td>
<td>Deletes HostingPowerAction Metadata from the HostingPowerAction objects</td>
</tr>
<tr>
<td>Remove-BrokerHypervisorAlertMetadata</td>
<td>Deletes HypervisorAlert Metadata from the HypervisorAlert objects</td>
</tr>
<tr>
<td>Remove-BrokerHypervisorConnection</td>
<td>Removes a hypervisor connection from the system.</td>
</tr>
<tr>
<td>Remove-BrokerHypervisorConnectionMetadata</td>
<td>Deletes HypervisorConnection Metadata from the HypervisorConnection objects</td>
</tr>
<tr>
<td>Remove-BrokerIcon</td>
<td>Remove an icon.</td>
</tr>
<tr>
<td>Remove-BrokerIconMetadata</td>
<td>Deletes Icon Metadata from the Icon objects</td>
</tr>
<tr>
<td>Remove-BrokerImportedFTA</td>
<td>Deletes one or more imported file type associations.</td>
</tr>
<tr>
<td>Remove-BrokerMachine</td>
<td>Removes one or more machines from its desktop group or catalog.</td>
</tr>
<tr>
<td>Remove-BrokerMachineCommand</td>
<td>Cancel a pending command queued for delivery to a desktop.</td>
</tr>
<tr>
<td>Remove-BrokerMachineCommandMetadata</td>
<td>Deletes MachineCommand Metadata from the MachineCommand objects</td>
</tr>
<tr>
<td>Remove-BrokerMachineConfiguration</td>
<td>Deletes a machine configuration from the site or removes the association from a desktop group.</td>
</tr>
<tr>
<td>Remove-BrokerMachineConfigurationMetadata</td>
<td>Deletes MachineConfiguration Metadata from the MachineConfiguration objects</td>
</tr>
<tr>
<td>Remove-BrokerMachineMetadata</td>
<td>Deletes Machine Metadata from the Machine objects</td>
</tr>
<tr>
<td>Remove-BrokerPowerTimeScheme</td>
<td>Deletes an existing power time scheme.</td>
</tr>
<tr>
<td>Remove-BrokerPowerTimeSchemeMetadata</td>
<td>Deletes PowerTimeScheme Metadata from the PowerTimeScheme objects</td>
</tr>
<tr>
<td>Remove-BrokerRebootCycleMetadata</td>
<td>Deletes RebootCycle Metadata from the RebootCycle objects</td>
</tr>
<tr>
<td>Remove-BrokerRebootSchedule</td>
<td>Removes the reboot schedule.</td>
</tr>
<tr>
<td>Remove-BrokerRemotePCAccount</td>
<td>Delete RemotePCAccounts from the system.</td>
</tr>
<tr>
<td>Remove-BrokerScope</td>
<td>Remove the specified catalog/desktop group from the given scope(s).</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Remove-BrokerSessionMetadata</td>
<td>Deletes Session Metadata from the Session objects</td>
</tr>
<tr>
<td>Remove-BrokerSiteMetadata</td>
<td>Deletes Site Metadata from the Site objects</td>
</tr>
<tr>
<td>Remove-BrokerTag</td>
<td>Removes a tag from the system or clears a tag to object association.</td>
</tr>
<tr>
<td>Remove-BrokerTagMetadata</td>
<td>Deletes Tag Metadata from the Tag objects</td>
</tr>
<tr>
<td>Remove-BrokerUser</td>
<td>Remove broker user objects from another broker object</td>
</tr>
<tr>
<td>Rename-BrokerAccessPolicyRule</td>
<td>Renames a rule in the site's access policy.</td>
</tr>
<tr>
<td>Rename-BrokerAppAssignmentPolicyRule</td>
<td>Renames an application rule in the site's assignment policy.</td>
</tr>
<tr>
<td>Rename-BrokerAppEntitlementPolicyRule</td>
<td>Renames an application rule in the site's entitlement policy.</td>
</tr>
<tr>
<td>Rename-BrokerApplication</td>
<td>Renames an application.</td>
</tr>
<tr>
<td>Rename-BrokerAssignmentPolicyRule</td>
<td>Renames a desktop rule in the site's assignment policy.</td>
</tr>
<tr>
<td>Rename-BrokerCatalog</td>
<td>Renames a catalog.</td>
</tr>
<tr>
<td>Rename-BrokerDesktopGroup</td>
<td>Renames a desktop group.</td>
</tr>
<tr>
<td>Rename-BrokerEntitlementPolicyRule</td>
<td>Renames a desktop rule in the site's entitlement policy.</td>
</tr>
<tr>
<td>Rename-BrokerMachineConfiguration</td>
<td>Renames a machine configuration.</td>
</tr>
<tr>
<td>Rename-BrokerPowerTimeScheme</td>
<td>Changes the name of an existing power time scheme.</td>
</tr>
<tr>
<td>Rename-BrokerTag</td>
<td>Rename one or more tags.</td>
</tr>
<tr>
<td>Reset-BrokerLicensingConnection</td>
<td>Resets the broker's license server connection.</td>
</tr>
<tr>
<td>Reset-BrokerServiceGroupMembership</td>
<td>Makes the broker load, from the specified Configuration Service instance, the addresses that the Service can be reached on.</td>
</tr>
<tr>
<td>Send-BrokerSessionMessage</td>
<td>Sends a message to a session.</td>
</tr>
<tr>
<td>Set-BrokerAccessPolicyRule</td>
<td>Modifies an existing rule in the site's access policy.</td>
</tr>
<tr>
<td>Set-BrokerAccessPolicyRuleMetadata</td>
<td>Creates/Updates metadata key-value pairs for AccessPolicyRule</td>
</tr>
<tr>
<td>Set-BrokerAppAssignmentPolicyRule</td>
<td>Modifies an existing application rule in the site's assignment policy.</td>
</tr>
<tr>
<td>Set-BrokerAppEntitlementPolicyRule</td>
<td>Modifies an existing application rule in the site's entitlement policy.</td>
</tr>
<tr>
<td>Set-BrokerApplication</td>
<td>Changes the settings of an application to the value specified in the command.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Set-BrokerApplicationInstanceMetadata</td>
<td>Creates/Updates metadata key-value pairs for ApplicationInstance</td>
</tr>
<tr>
<td>Set-BrokerApplicationMetadata</td>
<td>Creates/Updates metadata key-value pairs for Application</td>
</tr>
<tr>
<td>Set-BrokerAssignmentPolicyRule</td>
<td>Modifies an existing desktop rule in the site's assignment policy.</td>
</tr>
<tr>
<td>Set-BrokerAssignmentPolicyRuleMetadata</td>
<td>Creates/Updates metadata key-value pairs for AssignmentPolicyRule</td>
</tr>
<tr>
<td>Set-BrokerCatalog</td>
<td>Sets the properties of a catalog.</td>
</tr>
<tr>
<td>Set-BrokerCatalogMetadata</td>
<td>Creates/Updates metadata key-value pairs for Catalog</td>
</tr>
<tr>
<td>Set-BrokerConfigurationSlotMetadata</td>
<td>Creates/Updates metadata key-value pairs for ConfigurationSlot</td>
</tr>
<tr>
<td>Set-BrokerControllerMetadata</td>
<td>Creates/Updates metadata key-value pairs for Controller</td>
</tr>
<tr>
<td>Set-BrokerDBConnection</td>
<td>Specifies the Citrix Broker Service's database connection string.</td>
</tr>
<tr>
<td>Set-BrokerDesktopGroup</td>
<td>Adjusts the settings of a broker desktop group.</td>
</tr>
<tr>
<td>Set-BrokerDesktopGroupMetadata</td>
<td>Creates/Updates metadata key-value pairs for DesktopGroup</td>
</tr>
<tr>
<td>Set-BrokerEntitlementPolicyRule</td>
<td>Modifies an existing desktop rule in the site's entitlement policy.</td>
</tr>
<tr>
<td>Set-BrokerEntitlementPolicyRuleMetadata</td>
<td>Creates/Updates metadata key-value pairs for EntitlementPolicyRule</td>
</tr>
<tr>
<td>Set-BrokerHostingPowerAction</td>
<td>Changes the priority of one or more pending power actions.</td>
</tr>
<tr>
<td>Set-BrokerHostingPowerActionMetadata</td>
<td>Creates/Updates metadata key-value pairs for HostingPowerAction</td>
</tr>
<tr>
<td>Set-BrokerHypervisorAlertMetadata</td>
<td>Creates/Updates metadata key-value pairs for HypervisorAlert</td>
</tr>
<tr>
<td>Set-BrokerHypervisorConnection</td>
<td>Sets the properties of a hypervisor connection.</td>
</tr>
<tr>
<td>Set-BrokerHypervisorConnectionMetadata</td>
<td>Creates/Updates metadata key-value pairs for HypervisorConnection</td>
</tr>
<tr>
<td>Set-BrokerIconMetadata</td>
<td>Creates/Updates metadata key-value pairs for Icon</td>
</tr>
<tr>
<td>Set-BrokerMachine</td>
<td>Sets properties on a machine.</td>
</tr>
<tr>
<td>Set-BrokerMachineCatalog</td>
<td>Moves one or more machines into a different catalog.</td>
</tr>
<tr>
<td>Set-BrokerMachineCommandMetadata</td>
<td>Creates/Updates metadata key-value pairs for MachineCommand</td>
</tr>
<tr>
<td>Set-BrokerMachineConfiguration</td>
<td>Sets the properties of a machine configuration.</td>
</tr>
<tr>
<td>Set-BrokerMachineConfigurationMetadata</td>
<td>Creates/Updates metadata key-value pairs for MachineConfiguration</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Set-BrokerMachineMaintenanceMode</td>
<td>Sets whether the specified machine(s) are in maintenance mode.</td>
</tr>
<tr>
<td>Set-BrokerMachineMetadata</td>
<td>Creates/Updates metadata key-value pairs for Machine</td>
</tr>
<tr>
<td>Set-BrokerPowerTimeScheme</td>
<td>Modifies an existing power time scheme.</td>
</tr>
<tr>
<td>Set-BrokerPowerTimeSchemeMetadata</td>
<td>Creates/Updates metadata key-value pairs for PowerTimeScheme</td>
</tr>
<tr>
<td>Set-BrokerPrivateDesktop</td>
<td>Change the settings of a private desktop.</td>
</tr>
<tr>
<td>Set-BrokerRebootCycleMetadata</td>
<td>Creates/Updates metadata key-value pairs for RebootCycle</td>
</tr>
<tr>
<td>Set-BrokerRebootSchedule</td>
<td>Updates the values of one or more desktop group reboot schedules.</td>
</tr>
<tr>
<td>Set-BrokerRemotePCAccount</td>
<td>Modify one or more RemotePCAccounts.</td>
</tr>
<tr>
<td>Set-BrokerSession</td>
<td>Sets properties of a session.</td>
</tr>
<tr>
<td>Set-BrokerSessionMetadata</td>
<td>Creates/Updates metadata key-value pairs for Session</td>
</tr>
<tr>
<td>Set-BrokerSharedDesktop</td>
<td>Change the settings of a shared desktop.</td>
</tr>
<tr>
<td>Set-BrokerSite</td>
<td>Changes the overall settings of the current XenDesktop broker site.</td>
</tr>
<tr>
<td>Set-BrokerSiteMetadata</td>
<td>Creates/Updates metadata key-value pairs for Site</td>
</tr>
<tr>
<td>Set-BrokerTagMetadata</td>
<td>Creates/Updates metadata key-value pairs for Tag</td>
</tr>
<tr>
<td>Start-BrokerCatalogPvdImagePrepare</td>
<td>Start the PVD Image prepare process in the Broker for the machines</td>
</tr>
<tr>
<td>Start-BrokerMachinePvdImagePrepare</td>
<td>in the specified catalog(s).</td>
</tr>
<tr>
<td>Start-BrokerNaturalRebootCycle</td>
<td>Reboots all machines from the specified catalog when they are not in use.</td>
</tr>
<tr>
<td>Start-BrokerRebootCycle</td>
<td>Creates and starts a reboot cycle for each desktop group that contains machines from the specified catalog.</td>
</tr>
<tr>
<td>Stop-BrokerRebootCycle</td>
<td>Cancels the specified reboot cycle.</td>
</tr>
<tr>
<td>Stop-BrokerSession</td>
<td>Stop or log off a session.</td>
</tr>
<tr>
<td>Test-BrokerAccessPolicyRuleNameAvailable</td>
<td>Determine whether the proposed AccessPolicyRule Name is available for use.</td>
</tr>
<tr>
<td>Test-BrokerAppAssignmentPolicyRuleNameAvailable</td>
<td>Determine whether the proposed AppAssignmentPolicyRule Name is available for use.</td>
</tr>
<tr>
<td>Test-BrokerEntitlementPolicyRuleNameAvailable</td>
<td>Determine whether the proposed AppEntitlementPolicyRule Name is available for use.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>Test-BrokerApplicationNameAvailable</td>
<td>Determine whether the proposed Application Name is available for use.</td>
</tr>
<tr>
<td>Test-BrokerAssignmentPolicyRuleNameAvailable</td>
<td>Determine whether the proposed AssignmentPolicyRule Name is available for use.</td>
</tr>
<tr>
<td>Test-BrokerCatalogNameAvailable</td>
<td>Determine whether the proposed Catalog Name is available for use.</td>
</tr>
<tr>
<td>Test-BrokerDBConnection</td>
<td>Tests whether a database is suitable for use by the Citrix Broker Service.</td>
</tr>
<tr>
<td>Test-BrokerDesktopGroupNameAvailable</td>
<td>Determine whether the proposed DesktopGroup Name is available for use.</td>
</tr>
<tr>
<td>Test-BrokerEntitlementPolicyRuleNameAvailable</td>
<td>Determine whether the proposed EntitlementPolicyRule Name is available for use.</td>
</tr>
<tr>
<td>Test-BrokerLicenseServer</td>
<td>Tests whether or not a license server can be used by the broker.</td>
</tr>
<tr>
<td>Test-BrokerMachineNameAvailable</td>
<td>Determine whether the proposed Machine MachineName is available for use.</td>
</tr>
<tr>
<td>Test-BrokerPowerTimeSchemeNameAvailable</td>
<td>Determine whether the proposed PowerTimeScheme Name is available for use.</td>
</tr>
<tr>
<td>Test-BrokerRemotePCAccountNameAvailable</td>
<td>Determine whether the proposed RemotePCAccount OU is available for use.</td>
</tr>
<tr>
<td>Update-BrokerImportedFTA</td>
<td>Imports or updates all of the file type associations for the specified worker.</td>
</tr>
<tr>
<td>Update-BrokerNameCache</td>
<td>Performs administrative operations on the user and machine name cache.</td>
</tr>
</tbody>
</table>
Citrix Broker SDK - Access Policy

SHORT DESCRIPTION
Controls client-connection-based access to desktop groups.

LONG DESCRIPTION
The site's access policy defines rules controlling a user's access to desktop groups. Access checks are based on details of the user's connection from their user device to the site. Think of the access policy informally as a connection-based firewall.

The access policy comprises a set of rules. Each rule:

- Relates to a single desktop group.
- Contains a set of connection filters and access right controls.

Multiple rules can apply to the same desktop group.

By default, users have no access to any desktop group within a site. A user gains access to a group when their connection's details match the connection filters of one or more rules in the access policy.

The access policy also grants control rights over desktop and application sessions. For example, it can specify which protocols are allowed for a connection from a given endpoint, and whether the user can restart their machine.

To use a resource published by a site, the user must have both access to the desktop group that contains it, and an entitlement to use the resource. Entitlements are typically granted by the site entitlement and assignment policies; see about_Broker_Policies for more information.

ACCESS POLICY RULES
A single access policy rule relates to a single specified desktop group and comprises a set of connection filters and access right grants as described below.

Each rule can be individually enabled or disabled. A disabled rule is ignored when the access policy is evaluated.

CONNECTION FILTERS OVERVIEW
The connection filters in an access policy rule comprise the following:

- Local/remote client (SmartAccess) filters
- Client IP address filters
- Client name filters
- User filters
All filters have an include and exclude form that can be individually enabled or disabled. For a rule to be considered when the access policy is evaluated, at least one connection include filter must be enabled. By default, all filters, both include and exclude, are disabled.

The detailed behavior of connection filters is covered later.

ACCESS RIGHT CONTROLS OVERVIEW

The access right controls in an access policy rule comprise the following:

- Allowed protocols
- Whether machine restart, or programmatic session logoff, is allowed

The detailed behavior of access right controls is covered later.

DETAILS OF CONNECTION FILTERS

To gain access to a desktop group the user’s connection must match the filter criteria of at least one access policy rule for that group.

To match a rule, a connection must match all the rule’s enabled include connection filters and must not match any of the rule’s enabled exclude filters. That is, entries in exclude filters take priority.

Because all rules are evaluated independently, if an exclude filter match prevents a connection gaining access to a desktop group through one rule, the connection may still gain access to the same group through a different rule.

The filters are described in pairs below, but within a single rule a match against any exclude filter prevents a connection from gaining access through that rule irrespective of which include filters within the rule were also matched.

SMARTACCESS FILTERS

SmartAccess filters allow filtering based on whether the client is directly connected (for example over a local area network (LAN)) or through Access Gateway. For connections through Access Gateway further filtering can be performed based on the tags supplied from Access Gateway itself. The key properties of SmartAccess filters are:

- AllowedConnections (include filter: Filtered, NotViaAG, ViaAG)

  This property controls the behavior of the include filter. The default value is Filtered. The possible values are as follows:

  -- Filtered (default)

  The filter matches any user connection not through Access Gateway. In addition, the filter may match user connections through Access Gateway subject to the following: if the IncludedSmartAccessTags property is empty, any such connection matches. However, if the property is not empty at least one SmartAccess tag from the filter property must match a SmartAccess tag supplied with the user’s
connection.

-- NotViaAG

The filter matches only user connections not through Access Gateway. The contents of the IncludedSmartAccessTags property are ignored.

-- ViaAG

The filter matches only user connections through Access Gateway. If the IncludedSmartAccessTags property is empty, any such connection matches. However, if the property is not empty at least one SmartAccess tag from the filter property must match a SmartAccess tag supplied with the user's connection.

The IncludedSmartAccessTags property referred to above forms part of the include filter and is used if AllowedConnections is set to Filtered or ViaAG. It comprises a simple list of Access Gateway tags that are matched against those provided in the user's connection details.

o ExcludedSmartAccessTags (exclude filter)

A simple list of Access Gateway tags that are matched against those provided in the user's connection details. If any tag in the list matches one supplied with the user's connection, the user's connection does not match the access policy rule containing the filter.

The exclude filter has no setting corresponding to the AllowedConnections property so its behavior is determined only by the ExcludedSmartAccessTags property.

SmartAccess filters are typically used to control local (through a LAN) and remote (through Access Gateway) access to a site. A common model is to define two access policy rules rules for a group, one for local access and one for remote. The remote rule might impose restrictions on the user device having appropriate antivirus software installed, and potentially exclude certain user groups who would be allowed access over the corporate LAN (see USER FILTERS below).

CLIENT IP FILTERS

Client IP filters allow filtering based on the IP address of the user's device. The key properties of client IP filters are:

o IncludedClientIPs (include filter)

A simple list of numeric IP address ranges that are matched against the user device. The filter matches if the device address falls within any of the ranges in the list.
- ExcludedClientIPs (exclude filter)

   A simple list of numeric IP address ranges that are matched against the user device. If any entry matches the device address, the user's connection does not match the access policy rule containing the filter.

An IP address range in these filters can be specified as a simple IP address or as a range using a conventional subnet mask.

**CLIENT NAME FILTERS**

Client name filters allow filtering based on the name of the user's device. The key properties of client name filters are:

- IncludedClientNames (include filter)

  A simple list of names that are matched against the user device. The filter matches if the device name matches any value in the list.

- ExcludedClientNames (exclude filter)

  A simple list of device names that are matched against the user device. If any entry matches the device name, the user's connection does not match the access policy rule containing the filter.

Note: The form of the device name presented to the site depends on the site configuration. For example, by default in these filters you cannot use the form of the name presented by Web Interface.

**USER FILTERS**

User filters allow filtering based on the identity of the user. The key properties of user filters are:

- AllowedUsers (include filter: Filtered, AnyAuthenticated, Any)

  This property controls the behavior of the include filter. The default value is Filtered. The possible values are as follows:

  -- Filtered (default)

      The filter matches if the user's logon token contains one or more users or user groups matching those specified in the IncludedUsers property. The IncludedUsers property is a simple list of users or user groups and is used only when the AllowedUsers property is set to Filtered.

  -- AnyAuthenticated

      The filter matches any authenticated Microsoft Windows user. The
contents of the IncludedUsers property are ignored.

-- Any

The filter matches any user. The contents of the IncludedUsers property are ignored. In the current implementation this value is handled in the same way as AnyAuthenticated.

- ExcludedUsers (exclude filter)

A simple list of users or user groups. If any entry matches one in the user's logon token, the user's connection does not match the access policy rule containing the filter.

The exclude filter has no setting corresponding to the AllowedUsers property so its behavior is determined only by the ExcludedUsers property.

DETAILS OF ACCESS RIGHT CONTROLS

The access right controls of an access policy rule determine rights that the user has over any desktop or application session that they obtain from the rule's desktop group.

The rights apply only if the user's connection matches the connection filters of a rule, and only if the user also has an entitlement to a desktop or application session from the associated desktop group.

The following properties define the access rights:

- AllowedProtocols

A simple list of communication protocols over which connections can be made to resources published by the desktop group. For example, use this to restrict protocols with high bandwidth requirements to connections originating from a LAN.

- AllowRestart

For single-session power-managed machines, allows the user to restart the machine (the machine is powered off using the capabilities of its hypervisor). For multi-session machines the user's session is simply logged off.

For a given connection, if multiple rules result in access being granted to a session from a desktop group, the user's rights are the combined rights of all the rules that matched for that group. The allowed protocol lists from all the rules are combined, and the user is granted restart rights if any one rule has AllowRestart set.

SEE ALSO
about_Broker_Policies
about_Broker_AssignmentPolicy
about_Broker_EntitlementPolicy
New-BrokerAccessPolicyRule
Get-BrokerAccessPolicyRule
Set-BrokerAccessPolicyRule
Rename-BrokerAccessPolicyRule
Remove-BrokerAccessPolicyRule
New-BrokerAssignmentPolicyRule
New-BrokerEntitlementPolicyRule
New-BrokerAppAssignmentPolicyRule
New-BrokerAppEntitlementPolicyRule
Add-BrokerUser
Published applications allow your users to launch applications as if they were installed on their devices when in fact they are hosted remotely. The applications are normally launched in a seamless window, meaning that users see only the application window and not an additional desktop.

Published applications are hosted on either desktop operating systems or server operating systems. Applications are published to users and desktop groups. As such, conceptually they exist “on top” of desktop groups, which are themselves built on top of catalogs. See about_Broker_Concepts for more information on catalogs.

There are two types of applications:

- HostedOnDesktop - application runs on a remote machine and is displayed on the local client desktop.
- InstalledOnClient - application is installed and run on the local client machine and has its window overlaid on to a remote desktop.

HOSTING APPLICATIONS

There are three main ways of hosting an application: using a private single-session VDI desktop, a shared single-session VDI desktop, and on shared multi-session server operating systems.

An application hosted on a shared single-session VDI desktop, when launched, is hosted on a random machine within the desktop group. An application hosted on a private single-session VDI desktop ensures that when a user launches the application it is always hosted on the same machine. An application hosted on shared multi-session server operating systems ensures that when a user launches the application it is always hosted on one of the least loaded servers in the desktop group.

You control how the application is hosted based on the kind of desktop group that you choose. Shared desktop groups randomly select a machine, and these desktop groups can only be created from catalogs with a Random AllocationType. On the other hand, private desktop groups host the application on the same machine for that user every time, and these desktop group types can only be created from catalogs with a Permanent AllocationType.

USER ACCESS & ASSIGNMENT

Users are not assigned an application directly, but instead they are required to first have access to the desktop groups on which the applications are published through access policy rules.
With shared desktop groups, access to a published application also needs an application entitlement policy rule. An application entitlement policy rule defines the set of users who are allowed per-session access to desktops in a specified desktop group.

With private desktop groups, access to a published application also needs an application assignment policy rule. An application assignment policy rule defines the set of users who are allowed access to a single application session in a desktop group.

Users are assigned private machines in one of two ways: pre-assigned or assign-on-first-use. Pre-assigned means that individual user accounts have been specified for the machines within that desktop group. A single machine can only have a single user account (not a group account) assigned to it, and likewise a user can only be assigned a single machine within a desktop group.

Assign-on-first-use requires less configuration. Machines are assigned to users the first time they log on. Rather than allocating users directly to machines, instead users are assigned to the private desktop group, either through individual user accounts or through user group accounts. Then, when a user assigned to the desktop group logs on, a machine is automatically allocated to them.

USER VISIBILITY

Users can be further filtered by configuring the visibility filter on top of the application. This restricts the application to a subset of the users that were granted access by the access policy and entitlement/assignment policy rules on the desktop group.

MULTIPLE DESKTOP GROUPS

You can publish an application to multiple desktop groups, which have to be of the same kind. Generally, an application that is published only to private desktop groups is referred to as a "private application". An application that is only published to shared desktop groups is referred to as a "shared application".

LICENSING

Hosted applications need the appropriate licenses to exist on the license server to be functional.

NAMING

Applications have three names that identify them: the Name, BrowserName and the PublishedName. The Name is unique for each application, is not visible to the user, and is primarily used for administrative purpose. The BrowserName is unique across the entire site, and is primarily used internally. The PublishedName is not unique and is the name seen by end users who have access to this application.

When creating an application, you only need to specify the Name. If no BrowserName is specified one is automatically generated. If no PublishedName is specified by default it is the same as the Name.

The following special characters are not allowed in the BrowserName, PublishedName or the Name: \\;.: * ? = < > | [ ] ( ) " '

To change the PublishedName or BrowserName of an application you must use the Set-BrokerApplication cmdlet.

To change the Name of an application you must use the Rename-BrokerApplication cmdlet.

OPTIONAL
You can configure file-type associations for applications, so that if a user double-clicks a document icon on their device, a published application automatically starts. For more information, see the help for New-BrokerConfiguredFTA.

You can apply tags to applications as another convenient way to further organize (and search for) applications. For more information, see the help for New-BrokerTag.

**CMDLETS**

**New-BrokerApplication**

Creates an application for publishing after the needed desktop groups, access policy and entitlement/assignment policy rules have been created.

**Add-BrokerApplication**

Adds one or more applications to a desktop group.

**Get-BrokerApplication**

Gets one or more applications.

**Remove-BrokerApplication**

Deletes one or more applications or it can be used to delete just the association of an application to a desktop group.

**Rename-BrokerApplication**

Changes the Name of an application.

**Set-BrokerApplication**

Changes the settings of application objects, except their names.

**EXAMPLES**

**SHARED APPLICATION (SingleSession)**
You have created a catalog with a Random AllocationType and SingleSession SessionSupport. It contains two machines called worker1 and worker2, both in the ACME domain. You publish an application with shared hosting as follows:

C:\PS> Write-Host "Create a desktop group, and add machines to it"
C:\PS> $dg = New-BrokerDesktopGroup "Shared Application Group"
-DesktopKind Shared -DeliveryType AppsOnly -SessionSupport 'SingleSession'
C:\PS> $m1 = Get-BrokerMachine -MachineName "ACME\worker1"
C:\PS> $m2 = Get-BrokerMachine -MachineName "ACME\worker2"
C:\PS> Add-BrokerMachine $m1 -DesktopGroup $dg
C:\PS> Add-BrokerMachine $m2 -DesktopGroup $dg
C:\PS> Write-Host "Create access policy rule for desktop group"
-Enabled $true -DesktopGroupUid $dg.Uid -IncludedUserFilterEnabled $true
-AllowedProtocols @("HDX") -AllowedUsers AnyAuthenticated
C:\PS> Write-Host "Create application entitlement policy for desktop group"
-IncludedUsers 'ACME\Domain Users' -DesktopGroupUid $dg.Uid -Enabled $true
C:\PS> Write-Host "Create an application"
C:\PS> New-BrokerApplication -Name "Notepad" -PublishedName "Notepad"
-CommandLineExecutable "notepad.exe" -DesktopGroup $dg.Uid

In turn, this set of commands: creates a shared single session desktop group for applications delivery; adds two machines (from a catalog with a Random AllocationType and SingleSession SessionSupport) to the desktop group; creates access policy and application entitlement policy rules; creates an application; specifies its name, the executable, and links the application to the desktop group that will host it.

SHARED APPLICATION (MultiSession)
You have created a catalog with a Random AllocationType and MultiSession SessionSupport. It contains one machine called worker1 in the ACME domain. You publish an application with shared hosting as follows:

C:\PS> Write-Host "Create a desktop group, and add machines to it"
C:\PS> $dg = New-BrokerDesktopGroup "Shared Application Group"
-DesktopKind Shared -DeliveryType AppsOnly -SessionSupport 'MultiSession'
C:\PS> $m1 = Get-BrokerMachine -MachineName "ACME\worker1"
C:\PS> Add-BrokerMachine $m1 -DesktopGroup $dg
C:\PS> Write-Host "Create access policy rule for desktop group"
-Enabled $true -DesktopGroupUid $dg.Uid -IncludedUserFilterEnabled $true
-AllowedProtocols @("HDX") -AllowedUsers AnyAuthenticated
C:\PS> Write-Host "Create application entitlement policy for desktop group"
-IncludedUsers 'ACME\Domain Users' -DesktopGroupUid $dg.Uid -Enabled $true
C:\PS> Write-Host "Create an application"
C:\PS> New-BrokerApplication -Name "Notepad" -PublishedName "Notepad"
In turn, this set of commands: creates a shared multi session desktop group for applications delivery; adds one machine (from a catalog with a Random AllocationType and MultiSession SessionSupport) to the desktop group; creates access policy and application entitlement policy rules; creates an application; specifies its name, the executable, and links the application to the desktop group that will host it.

PRIVATE PRE-ASSIGNED APPLICATION
You have a catalog with a Permanent AllocationType and SingleSession SessionSupport. It contains two machines called worker1 and worker2, both in the ACME domain. You publish an application with private hosting using pre-assigned machines as follows:

```
C:\PS> Write-Host "Create a desktop group, and add machines to it"
C:\PS> $dg = New-BrokerDesktopGroup "Private App G1" -DesktopKind Private -DeliveryType AppsOnly -SessionSupport 'SingleSession'
C:\PS> $m1 = Get-BrokerMachine -MachineName "ACME\worker1"
C:\PS> $m2 = Get-BrokerMachine -MachineName "ACME\worker2"
C:\PS> Add-BrokerMachine $m1 -DesktopGroup $dg
C:\PS> Add-BrokerMachine $m2 -DesktopGroup $dg
C:\PS> Write-Host "Setting access policy rule for desktop group"
C:\PS> Write-Host "Pre-Assign users to the machines"
C:\PS> Add-BrokerUser "ACME\user1" -Machine $m1
C:\PS> Add-BrokerUser "ACME\user2" -Machine $m2
C:\PS> Write-Host "Create an application"
C:\PS> New-BrokerApplication -Name "Notepad" -PublishedName "Notepad" -CommandLineExecutable "notepad.exe" -DesktopGroup $dg.Uid
```

In turn, this set of commands: creates a private desktop group for applications delivery; adds two machines (from a catalog with a Permanent AllocationType and SingleSession SessionSupport) to the desktop group; creates access policy; creates two user objects; pre-assigns a user to each machine; creates the application; assigns users to it; and links the application to the desktop group that will host it.

PRIVATE, ASSIGN-ON-FIRST-USE APPLICATION
You have a catalog created with a Permanent AllocationType and SingleSession SessionSupport. It contains two machines called worker1 and worker2, both in the ACME domain. You publish an application with private hosting using assign-on-first-use machines as follows:

```
C:\PS> Write-Host "Create a desktop group, and add machines to it"
C:\PS> $dg = New-BrokerDesktopGroup "Private App G2" -DesktopKind Private
```
In turn, this set of commands: creates a private single session desktop group for applications delivery; adds two machines (from a catalog with a Permanent AllocationType and SingleSession SessionSupport) to the desktop group; creates access policy and application assignment policy rules; creates the application, and links the application to the desktop group that will host it.

SEE ALSO

about_Broker_Concepts

about_Broker_Desktops

New-BrokerApplication

Add-BrokerApplication

Remove-BrokerApplication

Rename-BrokerApplication

Set-BrokerApplication

New-BrokerAppAssignmentPolicyRule

Remove-BrokerAppAssignmentPolicyRule

Set-BrokerAppAssignmentPolicyRule

New-BrokerAppEntitlementPolicyRule

Remove-BrokerAppEntitlementPolicyRule

Set-BrokerAppEntitlementPolicyRule
Citrix Broker SDK - Assignment Policy

SHORT DESCRIPTION
Controls the automatic, permanent assignment of machines to users.

LONG DESCRIPTION
The site's assignment policy defines rules controlling automatic assignment of machines to users in a process referred to as Assign On First Use (AOFU).

In this assignment model, a desktop group is initially populated with machines that have no assignments, and users are granted entitlements to obtain a machine selected at random from the pool by self-service assignment. Once made, the assignment is permanent. The resulting assigned machines can be used to deliver either desktop or application sessions depending on the delivery type of the desktop group.

The assignment policy comprises a set of rules. The policy can be applied only to desktop groups of desktop kind Private.

For assignment of machines to deliver desktop sessions:

- Multiple rules can apply to the same desktop group.
- Each rule grants an entitlement to one or more machines.

For assignment of machines to deliver application sessions:

- Only a single rule can apply to a given desktop group.
- Each rule grants an entitlement to a single machine.
- Although only a single application assignment rule can be defined for a group, a user can still launch multiple applications from that group because the applications all run within the same session on the assigned machine.

Once a machine is assigned to a user by an assignment policy rule, the rule plays no further part in controlling access to that machine. The rule can be changed, or even removed, without impacting the user's access to the machine.

Rules for desktop and application machine assignments are distinct. Desktop assignments are managed through the BrokerAssignmentPolicyRule SDK object, and application rules through the BrokerAppAssignmentPolicyRule object.

Desktop machine assignment rules can be created only for desktop groups with delivery type DesktopsOnly, whereas an application machine assignment rule can be created only for delivery type AppsOnly.

Because desktop groups of assigned machines do not allow delivery type DesktopsAndApps, desktop machine assignment
Assignment policy rules are also used to configure the RemotePC feature where their detailed usage differs. For more information on the RemotePC feature see "help about_Broker_RemotePC".

For an entitlement granted by the assignment policy to be available to a user, the site's access policy must also grant them access to the desktop group.

ASSIGNMENT POLICY RULES

Each assignment policy rule has the following key properties:

- The desktop group to which it applies (one rule only ever applies to one group).
- The users to which machines can be assigned by the rule.

Additionally for desktop assignment rules, the following properties exist:

- The published name of the entitlement (visible to the user).
- The number of machines (entitlements) to which the rule grants access.
- The properties that a newly assigned desktop acquires.

If multiple desktop assignment rules entitle a user to machines from the same desktop group, their total machine entitlement is the sum of those granted by all those rules. The properties of the desktop sessions obtained may differ depending on which of the entitlements the user selects to start a session.

Each rule can be individually enabled or disabled. A disabled rule is ignored when the assignment policy is evaluated.

USER FILTERS (FULL)

Each rule has two user filters, an include filter and an exclude filter:

- The include filter contains users and user groups that are granted entitlements to machines.
- The exclude filter contains users and user groups that are denied entitlements to machines.

If the include filter of a rule contains multiple instances of a user (either explicit or implicit), this does not alter the number of machine entitlements granted to them by the rule.

Entries in the exclude filter take priority, so if a user appears explicitly or implicitly in both filters, access is denied. Typically, you use this filter to exclude a user or group of users who would otherwise gain access because they are members of a user group specified in the include filter.

Because all rules are independently evaluated, the exclude filter can exclude only users who would otherwise gain an entitlement through the same rule's include filter. That is, if a user is in a rule's include filter but not its exclude filter, the rule
is guaranteed to grant that user access to a machine irrespective of whether the user appears in the exclude filter of other rules.

If a filter contains a user group that contains other users and groups, the filter implicitly includes all of those users and groups.

By default the exclude filter is disabled.

To maintain assignment policy rules that can be fully displayed and edited with Citrix Studio, use the simplified user filter model below and do not use the exclude filter.

USER FILTERS (SIMPLIFIED)

The included user filter described above also supports a simplified usage model where the filter itself is disabled. When this is done, any user who has access to the desktop group through the access policy is implicitly granted an entitlement to a machine through the entitlement policy rule without the need to list the user in the rule’s include filter.

This is useful in cases where the access policy for the desktop group already explicitly specifies the users who should have access.

Even if the include filter is disabled, the exclude filter can still be used to deny the entitlement to users who would otherwise gain access through the access policy alone.

This simplified usage cannot be used for RemotePC desktop groups.

REMOTE PC USAGE

When a desktop group is configured as a RemotePC group, the usage of the assignment policy differs in the following ways:

- Only a single rule can apply to a given desktop group.
- The delivery type of the desktop group must be DesktopsOnly.
- The simplified user filter model cannot be used. Users or user groups must appear explicitly in the included user filter of the assignment policy rule.
- If the included user filter is disabled then RemotePC assignment for the group is also effectively disabled.

For more information on the RemotePC feature see “help about_Broker_RemotePC”.

ADDITIONAL DESKTOP ASSIGNMENT RULE PROPERTIES

Desktop assignment rules specify the following additional properties:

- PublishedName
- Description
- IconUid
- ColorDepth
- SecureIcaRequired
The published name, description, and icon UID properties apply to the desktop entitlement itself and determine the way in which the entitlement is presented to the user in, for example, StoreFront.

The color depth and secure ICA properties apply to the desktop session that is obtained when the entitlement is used.

In all cases, these properties can be explicitly specified. However, a null value (the default) means that the corresponding property is taken from the desktop group to which the rule applies. This inheritance from groups is dynamic; if the property of the group changes, the property of the entitlement changes too.

For assignment rules these properties are copied to the newly assigned desktop when the granted entitlement is first used. If the properties on the rule are subsequently changed the properties on the user's assigned desktop do not change. The dynamic inheritance of desktop properties from the desktop group continues after the assignment has occurred if the original rule did not provide explicit property values.

**CALCULATING OVERALL MACHINE ENTITLEMENTS**

Assignment rules are modified only by the SDK; they are not modified when used by the system to make automatic assignments. The number of machine entitlements offered to the user is determined by the number of machines (within the desktop group) already assigned to them and by which rules those assignments were made.

For each desktop group, the number of desktop entitlements available to the user is determined as follows:

1. The total number of entitlements for the user to machines in the group, from all rules, is calculated.

2. The total number of assigned machines (from any source) that the user already has in the group is determined.

3. The outstanding entitlement value is derived as the difference of the above two numbers.

4. If the outstanding entitlement value is zero or fewer, no further entitlements are allowed.

   Otherwise, for each applicable rule, the number of entitlements is that defined by the rule itself, minus the number of desktops already assigned to the user by that rule, and capped by the outstanding entitlement value.

Desktop and application machine entitlements both follow the above rules. However, because only a single application assignment rule granting a single entitlement per group can exist, these rules are seldom a consideration for applications.
EXAMPLES

Simple case:
A user is entitled to a single machine in a group by a single assignment rule:

1. On first use, the user sees a single desktop entitlement.
   If the user selects this, a new desktop is assigned.
2. On subsequent uses, the user sees the assigned desktop only.
   No other entitlements are displayed.

Complex case:
A user is entitled to two machines, one from each of two different rules, A and B, both applying to the same desktop group. In addition, the user has a machine explicitly assigned to them by the administrator within that group:

1. On first use, the user sees the administrator-assigned desktop, a single entitlement to a desktop of type A, and a single entitlement to a desktop of type B. If either of the entitlements is selected, a desktop of the appropriate type, A or B, is assigned.
2. On subsequent uses, the user sees the administrator-assigned desktop and the desktop assigned by the selected entitlement. No further entitlements are displayed.

MACHINE ENTITLEMENT PRESENTATION
For desktop machine assignment rules, the user interface determines whether users can visually distinguish between an entitlement to a machine and an actual assigned desktop. This cannot be controlled by Broker SDK cmdlets.

Although the number of entitlements seen by users takes account of all of their currently assigned desktops, the dynamic state of the system can affect the user interaction. This means that entitlements are displayed even where the pool of machines is empty, or the remaining desktops are in maintenance mode or otherwise unavailable. If the user attempts to use an entitlement in these cases, they receive a no-available-desktop error.

For application machine entitlement rules, the entitlements themselves are not presented to the end user; only the applications available to the user are presented. The use of available assignments, or of already assigned machines, is managed automatically.

NOTES
If an assignment rule grants entitlements to a user group:
The machine is assigned to the user who selects the entitlement, thus an assignment policy rule cannot be used to assign a machine to a user group. However, an administrator can assign a machine to a user group using the Add-BrokerUser cmdlet.

The number of machine entitlements specified in a desktop assignment rule applies to each member of the user group. For example, if a rule grants a user group access to two machines, every user in the group is entitled to two desktops.

The total number of machine entitlements defined by the assignment policy for a given desktop group may exceed the number of machines present in the group. A user attempting to use an entitlement when the pool of machines is empty receives a no-desktop-available error (See CALCULATING OVERALL MACHINE ENTITLEMENTS above).

A machine assigned to a user as a result of the assignment policy remains permanently assigned unless the machine assignment itself is removed by the administrator. Such assignments are permanent even if the assignment rule is deleted, when the machine is treated as administrator-assigned.

The assignment policy cannot be used to assign a machine to more than one user. This is only possible using the Add-BrokerUser cmdlet.

SEE ALSO

about_Broker_Policies
about_Broker_AccessPolicy
about_Broker_EntitlementPolicy
about_Broker_Applications
New-BrokerAssignmentPolicyRule
Get-BrokerAssignmentPolicyRule
Set-BrokerAssignmentPolicyRule
Rename-BrokerAssignmentPolicyRule
Remove-BrokerAssignmentPolicyRule
New-BrokerAppAssignmentPolicyRule
Get-BrokerAppAssignmentPolicyRule
Set-BrokerAppAssignmentPolicyRule
Rename-BrokerAppAssignmentPolicyRule
Remove-BrokerAppAssignmentPolicyRule
Add-BrokerUser
Citrix Broker - Concepts

Overview of the Citrix Broker.

The Citrix Broker is a Microsoft Windows service running on a delivery controller that responds to desktop/application launch requests from users through StoreFront by selecting a suitable machine, powering it up if necessary, and then returning the address of the selected machine to the user's endpoint system so that a session connection can be made. If required for resilience or scale, additional instances of the Broker may be installed to run on additional delivery controllers in the same delivery site.

In addition to handling launch requests, the Broker also has background responsibilities in the delivery site; these include: maintaining appropriate numbers of unused, powered-up machines to avoid unnecessary delays to users launching desktops/applications; maintaining periodic contact with powered-up machines; and monitoring the state of machines and user sessions.

The Citrix.BrokerAdmin PowerShell snap-in provides the cmdlets needed to administer and monitor the behaviour of the Broker service, either on the local system (by default) or on another system (by use of the -AdminAddress command-line parameter).

The cmdlets in the broker SDK manage objects in the following broad functional areas:

PROVISIONING CONFIGURATION

The Broker must be informed of the machines which are at its disposal. In order to do this, machines are organized in catalogs, created with the New-BrokerCatalog cmdlet; machines are introduced into the system through the use of the New-BrokerMachine cmdlet.

Catalogs define the nature of the machines contained within them, such as the allocation type (that is, static or random), how the machines are actually provisioned (that is, by MCS, PVS or manually), whether they are physical or virtual machines, whether they are single-session or multi-session, etc.

Typically, machines configured from a provisioning standpoint are not associated with specific users (though they may need to be, for example if the machine's disk image was captured from a specific user's physical desktop using a P2V tool); this is normally done when configuring how resources are brokered to users, below.

It is also possible for a catalog to be configured to be populated automatically with end users' existing physical machines using the RemotePC feature. The about_Broker_RemotePC topic gives more detail.

Provisioning configuration involves the following SDK objects:

BrokerHypervisorConnection
BROKERING CONFIGURATION

In order that resources (that is, desktops and applications) can be used in user sessions, the Broker must be configured to connect incoming user launch requests through StoreFront with the correct machine. This is achieved by adding machines to desktop groups. The grouping of machines in desktop groups need not necessarily match the grouping of the machines within the catalogs that were used for the configuration of provisioning. It is through the desktop group that the configuration of which users can use which machine resources is achieved.

Configuration of the mapping between resources and end users is achieved through a combination of machine assignment and entitlement rules. In addition, access to those resources must also be configured (for example, some resources could be configured only to be accessible to users when they are not connecting remotely through Access Gateway.) The about_Broker_Policies topic gives more detail of the rich configuration options available.

It is also possible for a desktop group to be configured to be populated automatically with end users existing physical machines using the RemotePC feature. The about_Broker_RemotePC topic gives more detail.

When machines are virtual, the broker can be configured to minimize power usage by switching them off when they are not expected to be in use while still maintaining good response times for end-user launch requests. This is achieved through power policy for desktop groups. This allows separate configuration for peak compared to off-peak times of the week of the number of machines nominally to be powered up, the number of machines to be powered up and idling, in addition to those in use to be used as a "buffer" to ensure prompt servicing of user launch requests, and the behavior required for virtual machines when users disconnect from their sessions for extended periods of time.

It is also possible to issue explicit power commands to machines. The about_Broker_PowerManagement topic gives more detail.

Configuration of Brokering involves the following SDK objects:

BrokerDesktopGroup
BrokerPrivateDesktop
BrokerSharedDesktop
BrokerRemotePCAccount
BrokerPowerTimeScheme
BrokerUser
BrokerTag
BrokerAccessPolicyRule
BrokerAssignmentPolicyRule
BrokerEntitlementPolicyRule
BrokerApplication
BrokerApplicationInstance
BrokerAppAssignmentPolicyRule
BrokerAppEntitlementPolicyRule
BrokerConfiguredFTA
BrokerImportedFTA

For more information, see:

Get-Help about_Broker_Desktops
Get-Help about_Broker_Policies
Get-Help about_Broker_Applications
Get-Help about_Broker_RemotePC
Get-Help Get-BrokerPrivateDesktop
Get-Help Get-BrokerSharedDesktop
Get-Help Get-BrokerPowerTimeScheme
Get-Help Get-BrokerUser
Get-Help Get-BrokerTag
Get-Help Get-BrokerAssignmentPolicyRule

MONITORING AND ADMINISTRATION

After you have provisioned and configured machines for brokering, use the broker SDK to monitor and administer user sessions and other aspects of the delivery site.

Monitoring and administration involve the following SDK objects:

BrokerServiceStatus
BrokerHypervisorAlert
BrokerDesktop
BrokerDesktopUsage
BrokerHostingPowerAction
BrokerSession
BrokerSessionMessage

For more information, see:
SITE MANAGEMENT

The broker must be configured after installation; this is normally performed automatically by the Citrix Studio console. Configuration tasks include selecting the database (and obtaining the SQL scripting to initialize it), selecting the Citrix Configuration Service that holds the site configuration.

Note that some aspects of broker configuration (such as the port number on which the broker listens for SDK connections) cannot be configured with PowerShell cmdlets. These are configured through the use of the Broker Service executable. For more information, see about_Broker_PostInstallPreConfiguration.

A further important aspect of site management concerns the way in which machines providing resources identify the delivery controllers to which they belong. For more information, see about_Broker_ControllerDiscovery.

Managing XenDesktop sites involves the following SDK objects:

- BrokerSite
- BrokerController
- BrokerDBConnection
- BrokerDBSchema
- BrokerDBVersionChangeScript
- BrokerInstalledDbVersion
- BrokerServiceInstance
- BrokerServiceGroupMembership
- BrokerNameCache

For more information, see:

- Get-Help about_Broker_PostInstallPreConfiguration
- Get-Help about_Broker_ControllerDiscovery
- Get-Help Get-BrokerSite
- Get-Help Get-BrokerController
- Get-Help Get-BrokerDBConnection
- Get-Help Get-BrokerDBSchema
- Get-Help Get-BrokerDBVersionChangeScript
- Get-Help Get-BrokerInstalledDbVersion
- Get-Help Get-BrokerServiceInstance
Get-Help Reset-BrokerServiceGroupMembership
Get-Help Update-BrokerNameCache
Collections of related settings may be applied to individual desktop groups through the creation of configuration slots and machine configurations.

A configuration slot defines a collection of related settings that are to be associated with that slot. Each machine configuration is associated with a single configuration slot and provides specific values for settings of that slot.

The SettingsGroup property of the configuration slot determines the particular collection of related settings that are associated with that slot. These groups are defined by Citrix and are not modifiable by administrators. For example, there is a particular group of Profile management specific settings that may be associated with a configuration slot. Because of the close association between a configuration slot and its collection of related settings, the full set of configuration slots is created during the site creation.

Each machine configuration is associated with a single configuration slot. The machine configuration contains policy data that provides specific values for the settings associated with that configuration slot.

Every value set in a machine configuration's policy must belong to the configuration slot's settings group. Therefore the appropriate SDK snap-in must be used to create the policy data. For example, the Profile management snap-in must be used to create the policy data for a machine configuration associated with the Profile management configuration slot.

To have particular policy settings applied to the machines in a desktop group, a machine configuration is associated with that desktop group. A machine configuration may be associated with multiple desktop groups. A desktop group may be associated with multiple machine configurations.

When a machine configuration is associated with a desktop group, the configuration inherits the delegated administration restrictions of the desktop group. Thus, if a machine configuration is associated with multiple desktop groups, an administrator can only modify the policy data of the configuration if the administrator has permission to modify every one of those desktop groups.

For detailed information about defining and assigning machine configurations, see:

- help New-BrokerMachineConfiguration
- help Add-BrokerMachineConfiguration

SEE ALSO

New-BrokerConfigurationSlot
New-BrokerMachineConfiguration
Add-BrokerMachineConfiguration
Import-BrokerDesktopPolicy
About Broker Controller Discovery

Oct 10, 2013

TOPIC

Citrix Broker - Configuring Controller Discovery

SHORT DESCRIPTION

Describes the way that machines providing published resources discover delivery controllers.

LONG DESCRIPTION

In order for the broker to be able to connect users to desktops and applications, the machines from which they are published must register (that is, establish communication) with the broker on an appropriate delivery controller in the delivery site.

The default operation, whose configuration is described in this topic, is to use information from the registry. This is referred to as registry-based controller discovery. The registry information can be supplied when installing the delivery agent software on each machine or it can be supplied through group-policy.

If machines are provisioned using quick deploy, information about delivery controllers is stored in a special “identity disk” attached to the VM.

Finally, in some deployments, the use of an Organizational Unit (OU) in Active Directory (AD) may be preferred. This is referred to as AD-based controller discovery. In this case, you must configure the GUID of the OU in the machines' registries. Such configuration is not described in this topic.

To perform registry-based controller discovery, run the PowerShell script called Set-ADControllerDiscovery.ps1 that is installed on each controller in the folder:

```
$Env:ProgramFiles\Citrix\Broker\Service\Setup Scripts
```

For more information, run this script with the -help parameter.
A desktop is a machine that is able to run a Microsoft Windows desktop environment (with a shell, icons and taskbar) or individual applications (seamlessly integrated with the local desktop). The configuration of the desktop determines whether it can run only desktop environments, only applications, or both desktops and applications. Machines running workstation operating systems are able to run one session at a time (single-session), whereas machines running server operating systems have the ability to run multiple simultaneous sessions (multi-session).

A key aspect of desktops is how they are assigned (or allocated) to users. Two allocation types are supported:

- **Random/Shared** - A user is assigned a desktop at random from a pool of shared desktops. Multi-session desktops are able to run sessions to multiple users simultaneously, whereas single-session desktops can only run one session at a time, and are returned to the pool when the user logs off. Single-session shared desktops usually discard user data stored on them after the user logs off. Multi-session shared desktops, however, do not tend to discard user data after a log-off, as this is only possible when the desktop is rebooted by a reboot schedule.

- **Permanent/Private** - A private desktop is permanently assigned to a specific user and data stored on it is retained across logons and restarts. A private desktop can have users assigned explicitly or on first use.

**DESKTOP GROUPS**

Desktops are collected together in desktop groups, and these provide a flexible grouping mechanism that can be used to associate:

- Desktops running on a particular type of machine
- Desktops with particular software installed
- Desktops for a set of users
- Desktops accessed in a similar way
- Desktops configured in a particular way
- Any combination of the above
Each desktop group can only contain one type of desktop, determined by its AllocationType and SessionSupport properties.

When assigning shared desktops or assign-on-first-use (AOFU) desktops to users, the set of candidates comes from available desktops in a particular desktop group.

You configure power management policy for single-session desktop groups, including peak and off-peak settings, for each desktop group. See about_Broker_PowerManagement for details.

CREATION OF DESKTOPS

Desktop objects are created automatically when a machine is added to a desktop group. The type of desktop is determined by the AllocationType property of the desktop group.

In order for a machine to be added from a catalog, the machine must be compatible with the desktop group. For this to be true, the catalog’s AllocationType must be compatible, and the SessionSupport property must match.

Note: Because the session support and functional level of the machine are determined by the software on the machine (operating system and Citrix VDA, respectively), the SessionSupport and MinimalFunctionalLevel of the catalog and desktop group may match, but not be compatible with the machine. In this case any attempt of registration by the machine will fail.

You can add machines explicitly using the Add-BrokerMachine cmdlet, or a number of free machines can be acquired from a catalog using the Add-BrokerMachinesToDesktopGroup cmdlet. A machine can only be associated with one desktop group at a time, and has a DesktopUid property that references the corresponding desktop object. You can also associate desktops to machines with their SID properties.

Desktop objects are deleted when the machine is removed from the desktop group (using the Remove-BrokerMachine cmdlet). Desktops are also deleted when the desktop group containing them is deleted (using the Remove-BrokerDesktopGroup cmdlet). A desktop cannot be removed from a catalog while it is in a desktop group.

SHARED (RANDOM) DESKTOPS

Shared desktops are published to users using entitlement policy rules. Each entitlement rule allows access to a single session on a desktop machine, selected at random from the available desktops in a desktop group (with a preference for desktops that are powered on). If there are no available desktops, launching the session fails. See about_Broker_EntitlementPolicy and about_Broker_PowerManagement for details.

PRIVATE (STATIC) DESKTOPS

You can assign private desktops to users explicitly or automatically, with the AOFU feature. It is also possible to assign private desktops to particular clients (through IP or client name).

You explicitly assign machines or desktops to users with the Add-BrokerUser cmdlet. Machines can be assigned to users before the machine has been added to the desktop group (desktop created), but otherwise the effect is the same.

With Add-BrokerUser you can assign a desktop to multiple users or user groups. If the desktop has single-session support then the desktop will be visible to multiple users, but only one user can log on to the desktop at any time.

Assignment policy rules allow you to use AOFU to assign desktops to users in a desktop group. When a user specified in an assignment policy rule launches a session, and if the user does not already have an assigned desktop, the broker selects an available desktop at random from the desktop group and permanently assigns it to that user. Once assigned in this way, the
desktop behaves as though the assignment was made explicitly by an administrator. See about_Broker_AssignmentPolicy for details.

It is possible to assign more than one desktop to a user. You can achieve this by explicit assignment, multiple assignment policy rules, or the MaxDesktops property of an assignment policy rule.

DESKTOP CONFIGURATION

When presented to the user, desktops are identified by:

- An icon (IconUid property)
- A name (PublishedName property)
- A description (Description property)

When starting the session, you can configure two connection settings:

- The color depth used at the start of the session (ColorDepth property)
- Whether SecureICA encryption is required (SecureIcaRequired property)

Each desktop group provides default values for these settings, but you can override them if the desktop has more specific settings (PrivateDesktop), or use an entitlement policy rule with more specific settings (SharedDesktop).

AOFU desktops can inherit these settings from the assignment policy rule when the assignment to the user takes place.

MAINTENANCE MODE

There are times when it is necessary to disable desktops. You can do this by setting the InMaintenanceMode property of a desktop to $true. This puts it into maintenance mode. The broker excludes single-session desktops in maintenance mode from brokering decisions and does not start new sessions on them. Existing sessions are unaffected. For multi-session desktops in maintenance mode, reconnections to existing sessions are allowed, but no new sessions are created on the machine.

Desktops in maintenance mode are also excluded from automatic power management, although explicit power actions are still performed.

Note that disabling desktop groups, entitlement policy rules, assignment policy rules, or applications are other ways of disabling aspects of brokering.

DESKTOP STATUS

Once desktops are created, you can query the configuration and state information for different kinds of desktop, or retrieve more information about desktops using the Get-BrokerMachine cmdlet. To get details of any sessions running on the desktops, use the Get-BrokerSession cmdlet.

You can also group desktops by a specific property, counting the number of desktops with each value using the Group-BrokerMachine cmdlet. This can provide useful summary statistics.

DESKTOP USAGE
Every hour the broker records how many desktops from each desktop group are in use, and the Get-BrokerDesktopUsage cmdlet returns this information. Analyze historical usage records to understand desktop usage patterns and help with the choice of idle pool and buffer settings.

**DESKTOP CONDITIONS**

CPU usage, ICA latency, and profile logon times of desktops are monitored. When one of these values exceeds a threshold (configured by policy), the condition is flagged in the DesktopConditions property of the desktop. When the value drops below the threshold again, the condition is cleared. Use Get-BrokerMachine or Group-BrokerMachines cmdlets to query this information.

**SEE ALSO**

- about_Broker_Concepts
- about_Broker_Applications
- about_Broker_EntitlementPolicy
- about_Broker_AssignmentPolicy
- Add-BrokerMachine
- Add-BrokerMachineToDesktopGroup
- Remove-BrokerMachine
- Add-BrokerUser
- Set-BrokerMachine
- Set-BrokerPrivateDesktop
- Get-BrokerMachine
- Group-BrokerMachine
- Get-BrokerDesktopUsage
About Broker Entitlement Policy

Oct 10, 2013

TOPIC

Citrix Broker SDK - Desktop and Application Entitlement Policy

SHORT DESCRIPTION

Controls end-user entitlement to desktop and application sessions provided from a pool of shared machines.

LONG DESCRIPTION

The site's entitlement policy defines rules controlling users' entitlements to desktop and application sessions from pools of shared machines. Each pool is defined by a desktop group.

The entitlement policy comprises a set of rules. Each rule grants users a single entitlement to a desktop or application session in a specified desktop group. The policy can be applied only to groups of desktop kind Random. For desktop entitlements multiple rules can apply to the same group, however for application entitlements only a single rule can apply to a given group.

When the user starts a session by selecting an entitlement the behavior depends on the session-support property of the desktop group:

- For single-session groups the user is temporarily assigned a machine selected at random from the group to provide their session. When the session ends, the machine is returned to the pool of available machines.
- For multi-session groups the user session is provided by the machine that is least loaded within the group when the session is launched.

If multiple desktop entitlement rules for the same group contain the same user, the user can have as many desktop sessions from the group concurrently as they have entitlements.

Although only a single application entitlement rule can be defined for a group, a user can still launch multiple applications from that group because the applications all run within that entitlement's single session.

Rules for desktop and application session entitlements are distinct. Desktop entitlements are managed through the BrokerEntitlementPolicyRule SDK object, and application rules through the BrokerAppEntitlementPolicyRule object.

Desktop entitlement rules can be created only for desktop groups with delivery types DesktopsOnly or DesktopsAndApps, whereas an application entitlement rule can be created only for delivery types AppsOnly or DesktopsAndApps.

For desktop groups with delivery type DesktopsAndApps, typically one or more desktop session entitlement rules together with a single application session entitlement rule exist.

For an entitlement granted by the entitlement policy to be available to a user, the site's access policy must also grant them access to the desktop group.

ENTITLEMENT POLICY RULES
Each entitlement policy rule has the following key properties:

- The desktop group to which it applies (one rule only ever applies to one group)
- The users to whom the entitlement is granted

Additionally for desktop entitlement rules, the following properties exist:

- The published name of the entitlement (visible to the user)
- Any properties that a desktop session launched using the entitlement should use that differ from the defaults specified on the desktop group

If multiple desktop entitlements are available to a user from the same group the resultant desktop session properties may differ depending on which entitlement the user selects to start the session.

Each rule can be individually enabled or disabled. A disabled rule is ignored when the entitlement policy is evaluated.

**USER FILTERS (FULL)**

Each rule has two user filters, an include filter and an exclude filter:

- The include filter contains users and user groups that are granted an entitlement to a session
- The exclude filter contains users and user groups that are denied an entitlement to a session

If the include filter of a rule contains multiple instances of a user (either explicit or implicit), they get only one entitlement by that rule.

Entries in the exclude filter take priority, so if a user appears explicitly or implicitly in both filters, access is denied. Typically, you use this filter to exclude a user or group of users who would otherwise gain access because they are members of a user group specified in the include filter.

Because all rules are independently evaluated, the exclude filter can only exclude users who would otherwise gain an entitlement through the same rule's include filter. That is, if a user is in a rule's include filter but not its exclude filter, the rule is guaranteed to grant that user a session entitlement irrespective of whether the user appears in the exclude filter of other rules.

If a filter contains a user group that contains other users and groups, the filter implicitly includes all of those users and groups.

By default the exclude filter is disabled.

To maintain entitlement policy rules that can be fully displayed and edited with Citrix Studio, use the simplified user filter model below and do not use the exclude filter.
USER FILTERS (SIMPLIFIED)

The included user filter described above also supports a simplified usage model where the filter itself is disabled. When this is done, any user who has access to the desktop group through the access policy is implicitly granted an entitlement to a session through the entitlement policy rule without the need to list the user in the rule's include filter.

This is useful in cases where the access policy for the desktop group already explicitly specifies the users who should have access.

Even if the include filter is disabled, the exclude filter can still be used to deny the entitlement from users who would otherwise gain access through the access policy alone.

ADDITIONAL DESKTOP ENTITLEMENT RULE PROPERTIES

Desktop entitlement rules specify the following additional properties:

- PublishedName
- Description
- IconUid
- ColorDepth
- SecureIcaRequired

The published name, description, and icon UID properties apply to the desktop entitlement itself and determine the way in which the entitlement is presented to the user in, for example, StoreFront.

The color depth and secure ICA properties apply to the desktop session that is obtained when the entitlement is used.

In all cases, these properties can be explicitly specified. However, a null value (the default) means that the corresponding property is taken from the desktop group to which the rule applies. This inheritance from groups is dynamic; if the property of the group changes, the property of the entitlement changes too.

NOTES

If a rule grants an entitlement to a user group, the session entitlement applies to the individual user who selects the entitlement. However, this does not prevent a different user in the same user group from using the same entitlement concurrently. So, a rule that grants an entitlement to a user group containing multiple users allows each user concurrent access to a single session from the desktop group.

The total number of entitlements defined by the policy may exceed the number of machines available, or the maximum allowed sessions, from the desktop group. A user attempting to use an entitlement when no further resources are available receives a no-desktop-available error.

If a session launched through an entitlement is active when the entitlement rule is deleted, the session continues unaffected. However:

- When the user ends the session, they cannot start a new one if the deleted rule was their only entitlement to a session in that group
- If the user disconnects the session, they cannot reconnect to it
SEE ALSO

about_Broker_Policies
about_Broker_AccessPolicy
about_Broker_AssignmentPolicy
about_Broker_Applications
New-BrokerEntitlementPolicyRule
Get-BrokerEntitlementPolicyRule
Set-BrokerEntitlementPolicyRule
Rename-BrokerEntitlementPolicyRule
Remove-BrokerEntitlementPolicyRule
New-BrokerAppEntitlementPolicyRule
Get-BrokerAppEntitlementPolicyRule
Set-BrokerAppEntitlementPolicyRule
Rename-BrokerAppEntitlementPolicyRule
Remove-BrokerAppEntitlementPolicyRule
Citrix Broker SDK - Error Handling

SHORT DESCRIPTION
Describes broker errors generated by cmdlets and how to access them.

LONG DESCRIPTION
The broker SDK cmdlets report errors through the class SdkErrorRecord, which is a subclass of the standard Powershell error record class System.Management.Automation.ErrorRecord. SdkErrorRecord contains:

- A short string to describe the error status code. This is implemented as a public property named Status.
- A dictionary of key-value pairs containing additional data specific to the cmdlet. This is implemented as a public property named ErrorData of type Dictionary<string, string>.

The error status property always has a value. Populating the error data dictionary is optional. The number of entries within the dictionary, the content of the entries, and the exact format of the key and value data is specific to each cmdlet.

You can access an SdkErrorRecord object using the standard Powershell cmdlet ErrorVariable parameter. The type of object returned by ErrorVariable depends on whether the error is terminating or non-terminating.

NON-TERMINATING ERRORS
For non-terminating errors, each object in the returned ErrorVariable array is simply an instance of type SdkErrorRecord.

TERMINATING ERRORS
For terminating errors, the object returned by ErrorVariable is of type System.Management.Automation.CmdletInvocationException.

For non-terminating errors that are escalated as terminating errors (through the "ErrorAction stop" argument), the object returned by ErrorVariable is of type System.Management.Automation.ActionPreferenceStopException.

CmdletInvocationException and ActionPreferenceStopException are subclasses of the base class System.Management.Automation.RuntimeException, which exposes the SdkErrorRecord object through its ErrorData property.

Class SdkOperationException
SdkErrorRecord's Exception property holds an instance of custom exception class SdkOperationException, which also contains the error status code and data dictionary from SdkErrorRecord.

The SDK cmdlets generate errors in response to exceptions generated by the underlying system or by the cmdlet detecting
errors locally and instantiating appropriate exception types. Such exceptions, which represent the original cause of the terminating error, are specified in SdkOperationException's InnerException property.

For terminating errors, use Powershell scripts to trap SdkOperationException and access additional error information and the originating exception.

**REVIEW OF POWERSHELL ERROR HANDLING BEHAVIOR**

Powershell scripts can access error information using the following methods:

- Read error records from the $error arraylist.
- Get the cmdlet to return error records using the standard ErrorVariable cmdlet parameter.
- Use trap blocks to catch exceptions for terminating errors.

The type of the error record that Powershell puts in the $error arraylist and returns through the ErrorVariable cmdlet parameter depends on whether the error is terminating or non-terminating, and whether the "ErrorAction continue" or "ErrorAction stop" cmdlet parameters are specified (that turn terminating errors into non-terminating ones, and vice versa).

For the combinations of error types (terminating, non-terminating) and error actions (stop, continue), the following statements describe the relationship between:

- The type of object contained in the Powershell $error arraylist.
- The type of object returned by the ErrorVariable cmdlet parameter (assuming the script can continue after a terminating error).
- The type of the exception that is thrown (where applicable).

Non-terminating errors with ErrorAction=Continue, $error type=SdkErrorRecord, and ErrorVariable type=SdkErrorRecord do not throw an exception.

Terminating errors with ErrorAction=Stop, $error type=ErrorRecord, and ErrorVariable type=CmdletInvocationException throw an SdkOperationException exception.

Non-terminating errors with ErrorAction=Stop, $error type=ErrorRecord, and ErrorVariable type=ActionPreferenceOperationException do not throw an exception.

Terminating errors with ErrorAction=Continue, $error type=ErrorRecord, and ErrorVariable type=CmdletInvocationException throw an SdkOperationException exception.


**ACCESSING ERROR RECORD DATA**

The Powershell code sample below demonstrates how to access error information programmatically with the ErrorVariable cmdlet parameter and a trap block.

```powershell
# Trap exceptions generated from terminating errors
trap [Exception] {
    # Code to handle exceptions
}
```
if($_.Exception.GetType().Name -eq "SdkOperationException")
{
    $sdkOpEx = $_.Exception
    # show error status
    write $("SdkOperationException.Status = " + $sdkOpEx.Status)

    # show error data dictionary
    write $("SdkOperationException.ErrorData=")
    write $("SdkOperationException.InnerException = " + $sdkOpEx.InnerException)
    $_.Exception.ErrorData
}
continue #could also call break here to halt script execution
write "TRAP BLOCK : END"

###########################################################################
## Run tests 1 to 4, below, in turn to examine terminating and non-terminating error behavior.
### Run tests 1 to 4, below, in turn to examine terminating and non-terminating error behavior.

# Test 1: Invoke cmdlet that generates a terminating error:
# New-BrokerCatalog throws terminating error if a catalog with the supplied name already exists.
#New-BrokerCatalog -Name "AlreadyExists" -AllocationType Random -ProvisioningType Manual -SessionSupport SingleSession -PersistUserChanges OnLocal -MachinesArePhysical $true -ErrorVariable ev

# Test 2: Force script execution to continue after a terminating error.
#New-BrokerCatalog -Name "AlreadyExists" -AllocationType Random -ProvisioningType Manual -SessionSupport SingleSession -PersistUserChanges OnLocal -MachinesArePhysical $true -ErrorVariable ev -ErrorAction continue

# Test 3: Invoke cmdlet that generates a non-terminating error:
# Get-BrokerCatalog generates a non-terminating error if a catalog with the specified name doesn't exist.
#Get-BrokerCatalog -Name "IDontExist" -ErrorVariable ev

# Test 4: Force script execution to halt after a non-terminating error.
#Get-BrokerCatalog -Name "IDontExist" -ErrorVariable ev -ErrorAction "stop"
write "" write "GET ERROR INFORMATION: BEGIN"
$SdkErrRecord = $null
if($ev[0].GetType().Name -eq "SdkErrorRecord")
{
$sdkErrRecord = $ev[0]

} elseif($ev[0].GetType().BaseType.FullName -eq "System.Management.Automation.RuntimeException") {

    $sdkErrRecord = $ev[0].ErrorRecord

} else {

    write ("UNKNOWN ERROR VARIABLE TYPE:"
    $ev[0].GetType().Name

}

if($sdkErrRecord -ne $null) {

    write ("Have sdkErrRecord:"
    write (" Type Name = " + $sdkErrRecord.GetType().FullName)
    write (" Status = " + $sdkErrRecord.Status)
    write (" Exception type = " + $sdkErrRecord.Exception.GetType().FullName)
    write (" ErrorData = ")
    $sdkErrRecord.ErrorData
    write (" FullyQualifiedErrorId = " + $sdkErrRecord.FullyQualifiedErrorId)

} write "GET ERROR INFORAMTION: END"
Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

```
Get-<Noun> | Where { $_.Size = 'Small' } | Sort 'Date' | Select -First 10
```

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

```
Get-<Noun> -Size 'Small' -SortBy 'Date' -MaxRecordCount 10
```

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

- **-MaxRecordCount <int>**
  
  Specifies the maximum number of results to return.
  For example, to return only the first nine results use:

  ```powershell
  Get-<Noun> -MaxRecordCount 9
  ```

  If not specified, only the first 250 records are returned, and if more are available, a warning is produced:
WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

To retrieve all records, specify a large number for -MaxRecordCount. As the value is an integer, you can use the following:

Get-<Noun> -MaxRecordCount [int]:MaxValue

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount

Get:<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get:<Noun> <<<< -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo : OperationStopped: (:) [Get:<Noun>], PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.
Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-sortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values. Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

-sortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'

-filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter 'Name -like "High*" -or (Priority -eq 1 -and Severity -ge 2)'

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

EXAMPLES

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company "citrix" -Product [X]EN"
Get-<Noun> -Product "Xen*" -Company "CITRIX"
Get-<Noun> -Filter { Company -eq 'Citrix' -and Product -like 'Xen*'}
See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
Get-<Noun> -Company "Abc[*]"                 # Matches Abc*
Get-<Noun> -Company "Abc**"                    # Matches Abc*
Get-<Noun> -Filter { Company -eq "Abc**" }    # Matches Abc*
Get-<Noun> -Filter { Company -eq "A""B""C" }   # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

```
Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```
Get-<Noun> -Filter 'Enabled'      # Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled' # Equivalent to 'Enabled -eq $false'
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```
Get-<Noun> -Shape [Shapes]:Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle" }
Get-<Noun> -Filter { Shape -like 'C*' }

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z" }
$d = [DateTime]"2010-08-23T12:30:00.0Z"
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge '-2' }  # Two days ago
Get-<Noun> -Filter { StartTime -ge '-1:30' }  # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge '-0:0:30' }  # 30 seconds ago

ARRAY PROPERTIES
When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings. Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*" }
Get-<Noun> -Filter { Users -contains "Fred*" }

You can also use the singular form with -Filter to search using other operators:

# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter { User -eq "Frederick" }
# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter { User -lt 'F' }

COMPLEX EXPRESSIONS
When matching against multiple values, you can use a sequence of comparisons joined with `-or` operators, or you can use `-in` and `-notin`:

```
Get-<Noun> -Filter { Shape -eq 'Circle' -or Shape -eq 'Square' }
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }
```

Braces can be used to group complex expressions, and override the default left-to-right evaluation of `-and` and `-or`. You can also use `-not` to invert the sense of any sub-expression:

```
Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue' -and Shape -eq 'Circle') }  
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue' -and Shape -eq 'Circle') }
```

**PAGING**

The simplest way to page through data is to use the `-Skip` and `-MaxRecordCount` parameters. So, to read the first three pages of data with 10 records per page, use:

```
Get-<Noun> -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 10 -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 20 -MaxRecordCount 10 <other filtering criteria>
```

You must include the same filtering criteria on each call, and ensure that the data is sorted consistently.

The above approach is often acceptable, but as each call performs an independent query, data changes can result in records being skipped or appearing twice. One approach to improve this is to sort by a unique id field and then start the search for the next page at the unique id after the last unique id of the previous page. For example:

```
# Get the first page
Get-<Noun> -MaxRecordCount 10 -SortBy SerialNumber

    SerialNumber ...  
    ---------------  ---  
    A120004          
    A120007          
    ... 7 other records ...  
    A120900          

# Get the next page
Get-<Noun> -MaxRecordCount 10 -Filter { FirstName -gt 'A120900' }

    SerialNumber ...  
```
FILTER SYNTAX DEFINITION

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= "{" <ComponentList> "}"

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> | <Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= "{" <ComponentList> "}" |

<PropertyName> <ComparisonOperator> <Value> | <PropertyName>

<AndOrOperator> ::= "-and" | "-or"

<NotOperator> ::= "-not" | "!

<ComparisonOperator>

 ::= "-eq" | "-ne" | "-le" | "-ge" | "-lt" | "-gt" |
  "-like" | "-notlike" | "-contains" | "-notcontains" | "-in" | "-notin"

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

  <scalar variable> | <array variable> |
  "$null" | "$true" | "$false"
Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30" means 1 hour and 30 minutes ago.
Citrix Broker - Licensing

Overview of broker licensing configuration.

As part of the licensing setup for a site, the types of product licenses used by the broker when creating connections to virtual desktops or applications can be configured using the Central Configuration Service SDK.

The following licensing related properties can be specified using the Set-ConfigSite cmdlet:

- **LicensingModel** - Sets the licensing model to use. Values can be ‘Concurrent’ or ‘UserDevice’.
- **ProductCode** - Specifies which product license is supported by the site. Values can be MPS for XenApp licenses or XDT for XenDesktop licenses.
- **ProductEdition** - Sets the licensing edition to use.

A license matching the specified model, product code, and edition must be available within the site's license server in order for the broker to grant licenses.

These properties are part of the site object returned by the Get-ConfigSite cmdlet.

**CONCURRENT LICENSE MODEL**

A concurrent license is tied to a XenDesktop session. When a user launches a session, a license is checked out to that session. When a user logs off from a session, the license is checked back in again, making it available for another session.

**USER DEVICE LICENSE MODEL**

With user device licensing, the license server automatically assigns licenses to users or devices based on usage:

- **User licensing** allows users to access their desktops and applications from multiple devices.
- **Device licensing** allows multiple users to access their desktops and applications from a single device.

When users or devices connect to an application or desktop, they consume a license for a 90-day license assignment period. The assignment period begins when a connection is made, is renewed to the full 90 days during the life of the connection, and expires (allowing reassignment) 90 days after the last connection terminates. A license assignment can be manually...
ended before the 90-day period elapses using the udadmin command line installed on the license server.

LICENSING STATE

The broker site contains the following properties related to licensing state:

- LicensingGracePeriodActive - Reports if the broker is in licensing grace period.
- LicensingOutOfBoxGracePeriodActive - Reports if the broker is in out-of-box grace period.
- LicensingGraceHoursLeft - The number of grace hours remaining, if the broker is in grace period, else it is null.
- LicensedSessionsActive - The number of active, licensed sessions.
- LicenseGraceSessionsRemaining - The number of grace sessions available, if the broker is in licensing grace period, else it is null.

These properties are part of the site object returned by the Get-BrokerSite cmdlet.

LICENSE SERVER TEST

The broker SDK cmdlet Test-BrokerLicenseServer checks whether or not a given server can be used as a license server by the broker.

RESETTING LICENSE SERVER CONNECTION

The broker SDK cmdlet Reset-BrokerLicensingConnection resets the broker's connection to the license server.

LICENSE BURN-IN DATE

The version of the product that is supported within the site is denoted by a licensing burn-in date. This date can be accessed through the LicensingBurnInDate field of the site object returned by the Get-ConfigSite cmdlet.

SEE ALSO

about_Broker_Site

Test-BrokerLicenseServer
Reset-BrokerLicensingConnection
Get-BrokerController
Set-ConfigSite
Get-ConfigSite
Get-BrokerSite
about_Broker_Machines

Oct 10, 2013

TOPIC

Citrix Broker SDK - Machine Object

SHORT DESCRIPTION

Describes machine concepts and usage.

LONG DESCRIPTION

A machine represents a physical or virtual machine that can be used to provide a user with one or more desktops, applications or both. When a machine is created, you must assign it to a catalog, which defines how the machine is allocated to a user (static or random), the session support it provides (single-session or multi-session), how the machine's disk images are created and managed (PVS, MCS or manually) and the expected functional level. If the machine is virtual but not provisioned by MCS, you must also assign it to a hypervisor connection, which represents the hypervisor (or pool of linked hypervisors) that runs the virtual machine.

Creating a machine object is the first step in the broker SDK towards configuring a physical or virtual machine to provide desktops and/or applications to users. A machine must be added to a desktop group before it is able to be used (see about_Broker_Desktops). To add machines to a desktop group, the Add-BrokerMachine or Add-BrokerMachinesToDesktopGroup cmdlets can be used. This creates a desktop object corresponding to the machine.

CATALOGS

When a machine is created, you must assign it to a catalog. The catalog defines the behavior of the machine within a site as well as the expected functionality and properties of the machine:

- Allocation type: The catalog determines how the machines are allocated to the user. Allocation can be static or random. Static allocation is where the machine is permanently assigned to a specific user. Data stored is retained across logons and restarts.

  The other type of allocation is random, where a random machine is assigned to a user from a pool when a session is requested. The machine returns to the pool when the user logs off.

- How the machine is created: The catalog collects together machines that are created in the same way: either with PVS, MCS or manually.

- Physical or Virtual: A machine that is virtual can have its power state controlled and monitored by the system. Virtual machines must be associated with a hypervisor connection, either directly or, in the case of MCS provisioned machines, indirectly through the provisioning scheme. Single session virtual machines can be managed using power policy to automatically be turned on or off as needed. Machines marked as physical are not monitored or controlled as to their power state.
- How the users settings are stored: The catalog also determines how the users settings are stored, either on a Citrix Personal vDisk, on the machine's local drive, or if the user settings are discarded.

- RemotePC: If the catalog is set up as a remote PC catalog, machines are added automatically upon registration based on the site configuration. In order for a catalog to be specified as a RemotePC catalog, the session support must be single session and the catalog must be set up for physical machines.

- Functional level: The functional level of a machine is determined by the version of the Citrix VDA software it is running. Some features are not supported in machines with lower functional levels. Catalogs can supply a minimal functional level, meaning any machines in the catalog with a lower functional level will be unable to register with the site.

- Session support: This can either be single-session or multi-session. Single-session machines can have an active session with up to one user at any time, whereas multi-session machines have the capability to have active sessions with multiple users simultaneously. The session support of a machine is determined by the variant of the VDA software component installed on the machine (either with single-session support or multi-session support). The multi-session VDA software may only be installed on server operating systems. The catalog session support must match the session support of the software installed on the machine for the machine to successfully register with the site.

MACHINE STATUS

After machines are created, you can query the configuration and state information using the Get-BrokerMachine cmdlet. The information the cmdlet can provide includes, but is not limited to, the following:

- Personal vDisk interactions and lifecycle: The current state of the personal vDisk can be obtained, as well as the configuration options of how the user data is persisted.

- Session properties: The properties of the current session for single-session machines can be obtained, such as ClientName and ClientAddress. To access session information on multi-session machines, the Get-BrokerSession cmdlet can be used.

- Application status: If the machine is configured to run applications, information can be found about the published applications running on the machine.

- Connection information: Information about the time and user of the last
connection to the machine can be found, as well as information about the last deregistration.

For an exhaustive list of the properties of a machine that can be queried, see the Get-BrokerMachine cmdlet.

**MACHINE CONFIGURATION**

Machine settings can be changed and configured once the machine object has been created, as long as the changes are compatible with the catalog the machine is in. For example, more users can be assigned to the machine than were initially assigned when creating the machine object, this is done with the Add-BrokerUser cmdlet.

For a full list of the machine configuration options available, see the Set-BrokerMachine command.

**MAINTENANCE MODE**

There are times when it is necessary to disable machines. This can be achieved by setting the InMaintenanceMode property to $true. This puts the machine into maintenance mode. With single-session machines, this means that the broker excludes the machine from brokering decisions and does not start new sessions on them. Existing sessions are unaffected. For multi-session desktops in maintenance mode, reconnections to existing sessions are allowed, but no new sessions are created on the machine.

Machines in maintenance mode are also excluded from automatic power management, although explicit power actions are still performed.

**SEE ALSO**

about_Broker_PowerManagement

about_Broker_Desktops

New-BrokerMachine

Add-BrokerMachine

Add-BrokerMachinesToDesktopGroup

Remove-BrokerMachine

Get-BrokerMachine

Set-BrokerMachine
**TOPIC**

Citrix Broker SDK - Access, Entitlement, and Assignment Policies

**SHORT DESCRIPTION**

Overview of the site policies that control users' access to desktop and application sessions.

**LONG DESCRIPTION**

For an end user to access a desktop or application resource within a site, they must have both an entitlement to use the resource, and have access to the desktop group that contains the resource.

Entitlements to use resources can be granted by one of the following means:

- The site entitlement policy grants entitlements to launch a shared desktop or application session from a pool of shared machines.
- The site assignment policy grants entitlements for "self service" permanent assignment of machines to users for running desktop or application sessions, and is referred to as "Assign On First Use" (AOFU)
- Machines can be permanently assigned ("pre-assigned") to users by the administrator to run either desktop or application sessions.
- Machines can be configured to allow automatic permanent assignment to their normal user (using the RemotePC feature).

A user must also be granted access to the desktop group that contains the resource. These access rights are controlled by the site's access policy.

The access policy controls access using details of the user's device such as whether it's connected over a local area network (LAN) or connected through Access Gateway, the user device's name, IP address or subnet, and the requested connection protocol. The user's identity can also feed into the access check allowing, for example, certain users access to resources only when locally connected to the site, but others full remote access.

Access and entitlements can be combined to allow rich and fine-grained control over which users have access to site resource from any given user device or location.

Each site has a single access policy, entitlement policy, and assignment policy. Each policy comprises a set of rules. Policies are defined by adding, removing, or changing rules.

Each site policy can also be viewed as a set of distinct policies each relating to a single desktop group. In general a group has one or more policy rules that relate to it, however each rule relates to only a single group. Thus the rules that grant entitlement and access rights to a desktop group define the policy for that group and that group only; changing this policy has no impact on the entitlement and access rights for any other other group in the site.

For detailed information about defining policy rules, see:
The mapping of policies to the resources that they make available within a site is described briefly below. For specific information on configuring each category of resource, consult the more detailed help topics listed.

**SHARED DESKTOP AND APPLICATION SESSIONS**

To grant access to a group of shared machines, use the access and entitlement policies:

- The access policy grants access to the desktop group containing the machines to be shared.
- The entitlement policy grants an entitlement to use one or more machines in the group to specified users or groups of users.

Groups of shared machines can be used to deliver full desktop or seamless application sessions, or both.

For more detailed information about configuring shared machines, see:

- [help about_Broker_AccessPolicy](https://docs.citrix.com)
- [help about_Broker_EntitlementPolicy](https://docs.citrix.com)

**PRE-ASSIGNED PRIVATE MACHINES**

To grant access to private machines, use the access policy and a machine assignment:

- The access policy grants access to the desktop group containing the machines.
- The assignment links the desktop to a specified user. You can assign a machine to just one user, multiple users or user groups. However, for single-session machines, only one user can access the machine at a time.

Private machines can be used to deliver full desktop or seamless application sessions (but not both).

For more detailed information about configuring private machines, see:

- [help about_Broker_AccessPolicy](https://docs.citrix.com)
- [help Add-BrokerUser](https://docs.citrix.com)
ASSIGN-ON-FIRST-USE (AOFU) MACHINES

To grant access to a desktop group containing assignable machines, use the access policy and the assignment policy:

- The access policy grants access to the desktop group containing the pool of machines.
- The assignment policy grants users a self-service entitlement to pick one or more machines from the pool.

AOFU machines can be used to deliver full desktop or seamless application sessions (but not both from the same desktop group).

For more detailed information about configuring AOFU desktops, see:

```
help about_Broker_AccessPolicy
help about_Broker_AssignmentPolicy
```

REMOTE PC MACHINES

The RemotePC feature allows existing physical machines to be assigned automatically to their normal user thus allowing them remote access to their own machine but without the need for the administrator to individually configure access to each machine.

For more detailed information about configuring the Remote PC feature, see:

```
help about_Broker_RemotePC
```

SEE ALSO

- about_Broker_AccessPolicy
- about_Broker_EntitlementPolicy
- about_Broker_AssignmentPolicy
- about_Broker_RemotePC
- New-BrokerAccessPolicyRule
- New-BrokerEntitlementPolicyRule
- New-BrokerAssignmentPolicyRule
- New-BrokerAppEntitlementPolicyRule
- New-BrokerAppAssignmentPolicyRule
Add-BrokerUser
Citrix Broker SDK - Post-Installation Configuration

SHORT DESCRIPTION
Describes how to configure the Citrix Broker Service port numbers, URL reservations, and Windows Firewall exclusions.

LONG DESCRIPTION
The XenDesktop installer configures the Citrix Broker Service with information specified during the installation. To change that configuration, use the BrokerService.exe command-line tool on each controller you want to change.

The default installation location of BrokerService.exe is:

   %ProgramFiles%\Citrix\Broker\Service\BrokerService.exe

BrokerService.exe supports the following optional command-line parameters:

-SdkPort <port> (default 80)

   Configures the port on which the broker listens for requests from SDK cmdlets. If you change this default value, specify the new value in the AdminAddress parameter on broker cmdlets. For example, if you changed the port to 8080, specify it as follows:

   Get-BrokerSite -AdminAddress localhost:8080

-VdaPort <port> (default 80)

   Configures the port on which the broker listens for registration requests from broker machines.

-WiPort <port> (default 80)

   Configures the port on which the broker listens for XML requests from StoreFront/Web Interface.

-WiSslPort <port> (default 443)
Configures the port on which the broker listens for SSL (Secure Socket Layer) XML requests from StoreFront/Web Interface.

-ConfigureFirewall

Configures Windows Firewall exclusions for the specified ports.

-Show

Shows the current configuration settings.

-Uninstall

Removes configuration settings, including Windows Firewall exclusions and URL reservations.

-LogFile <fileName>

Configures the file location for logging to a text file. The directory containing the log file must grant write access to the NetworkService account.

-Upgrade

Performs configurations required after an upgrade.

-Quiet

Suppresses console output for status messages.

-? or -Help

Shows usage information for the command.
Citrix Broker SDK - Machine Power Management

SHORT DESCRIPTION
Describes power management of machines used for desktops and applications.

LONG DESCRIPTION
The Citrix Broker Service is in day-to-day control of the power state of the configured desktop and application machines. The Broker Service can control several hypervisors, each hypervisor connection being handled by its own site service, so all Broker Service communication to the hypervisor is through one of the controllers in the site.

HYPERVISOR CONNECTIONS
Each hypervisor, or pool of linked hypervisors, is described and configured through the XdHyp pseudo-drive and associated Hyp PowerShell commands (cmdlets) which are provided by the host service snap-in. When you have first created the hypervisor connection using the host service cmdlets, you can create a broker equivalent object that references the Hyp instance using a GUID value. Use the broker's HypervisorConnection object to nominate a preferred controller for direct communication with the hypervisor on behalf of all other controllers for day-to-day power actions and status requests.

POWER ACTIONS AND THROTTLING
The site, through the Broker Service, can control the power state of the machines used by the site for desktops and applications. Power state changes can have a number of causes:

- Power policy rules, such as requests to shut down or suspend machines when user sessions on those machines end or are disconnected
- If allowed, user-driven desktop restarts
- Session launch requests requiring machines to be started
- Pool size management, which controls the number of running machines
- Direct administrator request using the SDK or Citrix Studio
- Reboot schedules and cycles
- Performing personal vDisk inventory activities
- Cleaning machines back to the golden master image state after they have been used

The power state changes of machines hosting desktops and applications are controlled using a queuing mechanism. Actions to change the power state are assigned a priority and are sent to the hypervisor according to a throttling mechanism. This avoids overloading the hypervisor.

The queuing and throttling mechanisms take place on a per-hypervisor-connection basis; each hypervisor connection's queue is dealt with independently. You can view the contents of the queues using the Get-BrokerHostingPowerAction cmdlet. This includes recently completed, in-progress, and pending actions (that is, those due to be sent to the hypervisor)
based on the throttling settings).

Each power action object comprises:

- The machine to be acted on (Name, DNS Name, Hosting Name)
- The action to be performed (TurnOn, TurnOff, Shutdown, Reset, Restart, Suspend, or Resume)
- The action's priority (Base, the original priority, and Actual, the current priority)
- The action's state (Pending, Started, Completed, Failed, Canceled, Deleted, or Lost)
- Time stamps of the action's lifecycle points (when it was created, started, or completed)
- Any reason the action failed

The throttling of power actions is controlled by three metadata values on the Hyp hypervisor connection object when accessed through the XdHyp pseudo-drive. The four values throttle power actions according to:

- The maximum absolute number of in-progress power actions
- The maximum number of in-progress power actions expressed as a percentage of the total number of machines controlled by the hypervisor connection
- The maximum number of new power actions sent to the hypervisor per minute
- The maximum number of in-progress PvD inventory activities expressed as a percentage of the total number of machines controlled by the hypervisor connection

You add power actions to the queue using the SDK's New-BrokerHostingPowerAction cmdlet. You cancel power actions in the queue using the Remove-BrokerHostingPowerAction cmdlet. You boost or reduce their priority using the Set-BrokerHostingPowerAction cmdlet.

You can also schedule power actions to be executed in the future, using the New-BrokerDelayedHostingPowerAction command. Only Shutdown and Suspend actions can be scheduled in this way. You can view these delayed power actions using Get-BrokerDelayedHostingPowerAction and cancel them with Remove-BrokerDelayedHostingPowerAction. When a delayed power action is ready to be executed it is deleted, and a corresponding normal power action is placed in the queue described above.

**POWER POLICY**

Policy rules associated with a desktop group allow you to change power states at configurable times after session state changes, typically a set number of minutes after session disconnection or session logout.

Note that these policy rules are defined directly as properties of the desktop group.

These policy actions allow the following operations to be specified:
o A power action to be performed at a defined period after a session is disconnected
o A power action to be performed at a defined extended period after a session is disconnected
o A power action to be performed at a defined period after a session is logged off

The two disconnect policy actions are designed to allow multi-stage policies such as initially suspending a machine shortly after a session disconnect occurs, and then later powering-off the machine if the session has not been reconnected.

At the set time after the session state change, the required action is added to the power action queue, and this is then throttled and processed as normal.

POOL SIZE MANAGEMENT

You can manage flexibly the number of machines running desktops and applications using the pool size. For any given hour of the day and day of the week, this is an absolute number of machines or the percentage of the total number of machines in the desktop group. The pool size specifies the total number of machines that are always running, regardless of whether they are in use or idle. (Note: The number of machines does not depend on their idle status, but this does affect the buffer size value, which is also used to manage pool sizes.)

To start or shut down desktop machines to achieve the desired pool size, the system places power actions in the queue. Standard throttling queue management sends these to the hypervisors. A single desktop group (and its pool) can span multiple hypervisors, so actions to start and shut down machines can be added to multiple queues.

POWER TIME SCHEMES

Each single-session desktop group can be associated with one or more power time schemes, each scheme covering a number of days of the week. The time schemes specify, for each hour of the day, whether that hour is peak or off-peak. They also specify the number of running unassigned machines maintained by the broker.

You can configure other settings, such as buffer size and any power policy rules differently for peak and off-peak hours. You can define the number of running machines, idle or in use, either as an absolute value or as a percentage of the desktop group size. Machines running desktops and applications are started (or shut down when not in use) to match the required pool size.

Each power time scheme comprises:

- The name of the scheme
- The pattern of days of the week covered by the scheme
- The set of hours considered peak and off peak
- The set of pool size values (one for each hour of the day)

The hours of the day used by time schemes are the hours in the time zone for the desktop group the scheme is associated with. You cannot associate one desktop group with multiple time schemes covering the same day of the week.

BUFFER SIZE
In addition to the pool size, you can optionally configure two buffer sizes for each desktop group, one for peak hours and one for off-peak hours. The buffer size defines the minimum number of idle unassigned machines maintained by the broker and is specified as a percentage of the total machines in the group. These are running machines that are not used by any user session. The buffer size on its own never causes machines to shut down. It causes them to start up so a minimum number of idle machine is always available. The buffer size in conjunction with the pool size can cause machines running desktops or applications to be shut down.

POWER MANAGEMENT OF ASSIGNED MACHINES

Automatic power management for private desktop groups provides the ability to power on all assigned machines at the transition to a peak period and respectively power off all machines at the transition to an off-peak period.

If a machine is shut down during peak hours it will not be automatically powered on again, unless the AutomaticPowerOnForAssignedDuringPeak property on the desktop group is also enabled.

Note that all power management facilities apply only to single session machines.

REBOOT SCHEDULES

Reboot schedules are commonly used after image updates or to perform regular reboots of all machines in a desktop group or catalog to clear down problems resulting from a corrupt state or hung/faulty applications.

Reboot schedules allow distributing the reboot operation of all machines over a provided duration. Individual machine reboots are scheduled in a way that attempts to maintain maximum availability of machines in the group as the reboots occur, and avoid boot storms that overload the underlying infrastructure.

Reboot schedules are the only form of automatic power management that can shut down a machine while users are logged on; however the administrator can provide a warning message to be displayed to end users at a specified period prior to the shutdown taking effect.

REBOOT CYCLES

Reboot cycles describe the dynamic execution of desktop group or catalog reboot operations. Reboot cycles can be created due to reboot schedules, or by on-demand reboot operations requested via the SDK.

RebootCycle objects encapsulate the details of the associated reboot operation and can be queried to show the current status.

STATUS

You can view the status of the hypervisor connection on the broker hypervisor connection object. You can obtain any hypervisor alerts using the Get-BrokerHypervisorAlert cmdlet. You can check the power state of machines running desktops or applications using the relevant Machine or Desktop objects.

SEE ALSO

about_Broker_Concepts
about_Broker_Machines
about_HypHostSnapin
New-BrokerHypervisorConnection
Get-BrokerHypervisorConnection
Set-BrokerHypervisorConnection
Remove-BrokerHypervisorConnection
New-BrokerHostingPowerAction
Get-BrokerHostingPowerAction
Set-BrokerHostingPowerAction
Remove-BrokerHostingPowerAction
New-BrokerDelayedHostingPowerAction
Get-BrokerDelayedHostingPowerAction
Remove-BrokerDelayedHostingPowerAction
New-BrokerPowerTimeScheme
Get-BrokerPowerTimeScheme
Set-BrokerPowerTimeScheme
Rename-BrokerPowerTimeScheme
Remove-BrokerPowerTimeScheme
Get-BrokerRebootCycle
Set-BrokerRebootCycle
Start-BrokerRebootCycle
Stop-BrokerRebootCycle
Get-BrokerRebootSchedule
Set-BrokerRebootSchedule
New-BrokerRebootSchedule
Remove-BrokerRebootSchedule
New-BrokerDesktopGroup
Get-BrokerDesktopGroup
Set-BrokerDesktopGroup
Add-HypMetadata
Remove-HypMetadata
Remote PC allows automatic publishing of a user's physical desktop within a XenDesktop site, so that it can be accessed remotely. The configuration of Remote PC within the Citrix Broker service specifies the rules and relationships that allow the machine to successfully register with a controller in the site, and be made available for the user to start remote sessions.

When Remote PC is configured, there are two steps required for publishing a machine as a Remote PC desktop.

First, a VDA must be installed on the machine and it must be configured such that it registers with a controller in a site.

Second, a user must log on to the machine. When the Citrix Broker service detects an active console session for a user, it assigns the user to the machine and publishes it in a Remote PC desktop group.

Remote PC configuration consists of defining relationships between:

- Machines and catalogs
- Catalogs and desktop groups
- Desktop groups and assignment policy rules
- Assignment policy rules and users

The Citrix Broker service automates the assignment of Remote PC machines to users in two stages. The first stage automatically imports matching unconfigured machines into the site:

- Suitable machines are automatically added to a Remote PC catalog.
- The machines are temporarily configured to be in one of the Remote PC desktop groups associated with the catalog.

When a suitable user logs on to the console of the machine, the second stage of the automatic configuration is performed:

- The machine is configured to be in the desktop group that is appropriate for the user.
- The machine is assigned to the user that has logged on.
The machine desktop is made available to the user remotely, configured to appear as the machine hostname.

Catalogs and desktop groups can be marked as participating in RemotePC automation with the 'IsRemotePC' property, but this property can only be set to true if various other properties of the catalog or desktop group are appropriate. Catalogs must be single-sessioned and contain physical machines. Desktop Groups must be single session, configured to deliver private assigned machines, delivering desktop sessions only.

Catalogs and desktop groups can have the 'IsRemotePC' property cleared to remove them from the RemotePC automation, but only when all RemotePC associations relating to them through RemotePCAccounts and catalog/group relationships have first been removed.

MACHINES AND CATALOGS

Mappings are defined between machines and Remote PC catalogs through the RemotePCAccount cmdlets.

The machine to catalog mappings support the automated addition of machines to catalogs.

PROPERTIES

RemotePCAccounts expose sets of included and excluded machine name filters specified in DOMAIN\MACHINE format. A MachinesIncluded or MachinesExcluded entry can include asterisk wildcards to generalize matches.

Each RemotePCAccount can specify the Distinguished Name (DN) of an AD container in addition to the machine name filters, in the RemotePCAccount Organisational Unit (OU) property, and this limits the machines that the account objects act on to those that reside at or below that container in the AD domain hierarchy. An AllowSubfolderMatches setting on the RemotePCAccount indicates whether the computer must exist directly within the container to trigger a match, or whether it can be in a child below the defined container in the AD hierarchy.

Note that the AD container component is optional and a special value of 'any' can be supplied in the OU field to permit the RemotePCAccount to automatically match regardless of the AD machine object location in the AD domain hierarchy. A match is still subject to machine name filtering.

The last component of the RemotePCAccount is the CatalogUid. This indicates which catalog the Remote PC automation should move the machine into when a match is found.

CONSTRAINTS

The IsRemotePC setting must be enabled on catalogs before they can be specified in a RemotePCAccount.

There can be any number of RemotePCAccounts configured in the site as long as each specifies a unique OU. There can be only one RemotePCAccount with the 'any' OU.

AUTOMATION

When a machine matching the criteria set up in a RemotePCAccount instance registers with one of the brokers in the site, it is automatically added to the catalog defined by the RemotePCAccount. This can take up to 30 seconds to happen after the machine registers.

When the machine registration occurs, the machine details can match more than one RemotePCAccount instance, but the machine can only be placed into one catalog, so one RemotePCAccount instance is chosen from the list that best matches
the machine. This choice is made according to the length in nodes of the DN of the AD container specification associated with the RemotePCAccount, so more specific child OUs override specifications for their parent OUs if both are present. The RemotePCAccount for the 'any' OU is always last and used only if no other instances match.

A Windows eventlog message is generated when an automated catalog assignment is performed.

NOTES

The AD distinguished name for the container is checked when the RemotePCAccount is created, but if the container is subsequently moved or deleted, the site does not automatically accommodate this, and the RemotePCAccount must be changed or removed manually.

Related Cmdlets

- New-BrokerRemotePCAccount
- Get-BrokerRemotePCAccount
- Set-BrokerRemotePCAccount
- Remove-BrokerRemotePCAccount
- New-BrokerCatalog [-IsRemotePC <Boolean>]
- Set-BrokerCatalog [-IsRemotePC <Boolean>]

CATALOGS AND DESKTOP GROUPS

A Remote PC catalog may be associated with one or more Remote PC desktop groups. The catalog to desktop group associations support automated publishing of machines to users.

USAGE

An association is formed via:

```
Add-BrokerDesktopGroup -RemotePCCatalog <Catalog> [-Priority <Int32>]
```

An association is broken via:

```
Remove-BrokerDesktopGroup -RemotePCCatalog <Catalog>
```

To find associated desktop groups:

```
Get-BrokerDesktopGroup -RemotePCCatalogUid <Int32>
```

AUTOMATION

When a machine in a Remote PC catalog is not already assigned to a user, Remote PC automation temporarily places the
machine in one of the desktop groups associated with the catalog. The temporary placing of the machine in a group will be adjusted as needed when the machine assignment to a user is first made.

The desktop group chosen as the temporary home for a machine is according to the priority value supplied when making the association between the group and the catalog. The group with the highest priority (lowest numerical priority value) is chosen.

A Windows eventlog message is generated when an automated machine assignment is performed.

PRIORITY

Each desktop group to catalog association has a priority value that can be specified when the association is made or defaults to lower than the lowest existing association priority (highest numerical value) or zero if no other association exists yet. The lower the priority numerical value, the higher the priority level. The priority value for the association is used to choose the temporary desktop group to place a new RemotePC machine in when it first registers. It is also used to decide which group to finally place the machine in when the user that the machine is being assigned to is appropriate for more than one of the associated desktop groups, usually because the desktop groups are using AD security groups for the user associations with the desktop groups.

The priority values for each association between a desktop group and a catalog are automatically maintained as unique for each catalog, and priorities can be adjusted up or down accordingly by the system. The last association created without a specified priority is always arranged to have the least priority (the highest numerical priority value).

NOTES

The temporary placement of the machines in a desktop group is not re-evaluated if settings change after the automatic placement.

RELATED CMDLETS

- Add-BrokerDesktopGroup -RemotePCCatalog <Catalog>
- Remove-BrokerDesktopGroup [-RemotePCCatalog <Catalog>]
- Get-BrokerDesktopGroup [-RemotePCCatalogUid <Int32>]
- New-BrokerCatalog [-IsRemotePC <Boolean>]
- Set-BrokerCatalog [-IsRemotePC <Boolean>]
- New-BrokerDesktopGroup [-IsRemotePC <Boolean>]
- Set-BrokerDesktopGroup [-IsRemotePC <Boolean>]

DESKTOP GROUPS, ASSIGNMENT POLICY RULES, AND USERS

Assignment policy rules are used to define sets of users allowed to be assigned to machines by Remote PC automation, and to determine which desktop group the machine is placed in when the user assignment is made.

AUTOMATION

A machine is automatically assigned to a user when the Citrix Broker service sees that the user has logged on to a RemotePC machine, and the user is amongst those configured in the desktop group assignment policy rule. If this is the first assignment of a user to a machine, all the assignment policy rules for all groups associated with the machine are checked,
and the machine can be moved to a different, more appropriate desktop group if needed.

A Windows eventlog message is generated when an automated assignment of a machine to a user is made.

MULTIPLE ASSIGNMENTS

By default, multiple automatic assignments of users to the same machine can be established if multiple users log on to the RemotePC machine, but this can be disabled if desired using a registry setting (see CTX137805).

NOTES

User to machine assignments can still be made and removed manually through the Powershell SDK for machines in Remote PC catalogs and desktop groups.

The automatic placement of a machine in an appropriate desktop group happens when the first automatic assignment of a user to the machine is made, and this placement will not be automatically updated if the configuration subsequently changes.

Only a single assignment policy rule is allowed for each RemotePC desktop group.

RELATED CMDLETS

- New-BrokerUser
- New-BrokerAssignmentPolicyRule
- Set-BrokerAssignmentPolicyRule

EXAMPLE

The following example creates a simple configuration that allows any user and machine in the current AD domain to participate in RemotePC.

```powershell
# Create a Remote PC catalog
$catalog = New-BrokerCatalog -IsRemotePC $true -SessionSupport SingleSession -MachinesArePhysical $true -AllocationType Static -PersistUserChanges OnLocal -ProvisioningType Manual -Name RemotePCCatalog

# Create a Remote PC desktop group
$dg = New-BrokerDesktopGroup -IsRemotePC $true -SessionSupport SingleSession -DeliveryType DesktopsOnly -DesktopKind Private -Name RemotePCDesktopGroup

# Create an assignment policy rule for that desktop group allowing any # domain user to match.
```

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.710
New-BrokerAssignmentPolicyRule -DesktopGroupUid $dg.Uid
   -IncludedUsers 'domain users'
   -Name RemotePCAPR

# Create a RemotePCAccount matching any unconfigured machine, causing the
# machines to be added to the catalog by Remote PC automation.
New-BrokerRemotePCAccount -OU 'any'
   -CatalogUid $catalog.Uid

# Associate the desktop group and catalog to permit domain users to be
# automatically assigned to machines in that catalog
Add-BrokerDesktopGroup $dg -RemotePCCatalog $catalog

# Create an access policy rule, allowing access to the users to the
# Remote PC desktop group.
New-BrokerAccessPolicyRule -IncludedUsers 'domain users'
   -DesktopGroupUid $dg.Uid
   -IncludedUserFilterEnabled $true
   -Name RemotePCAccessPolicyRule
Add-BrokerApplication

Oct 10, 2013
Adds applications to a desktop group.

Syntax


Add-BrokerApplication [-Name] <String> [-DesktopGroup <DesktopGroup>] [-Priority <Int32>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Add-BrokerApplication cmdlet is used to associate one or more applications with an existing desktop group.

There are two parameter sets for this cmdlet, allowing you to specify the application either by its BrowserName or by an array of object references. Uids can also be substituted for the object references.

See about_Broker_Desktops and about_Broker_Applications for more information.

Related topics

New-BrokerApplication
Add-BrokerApplication
Add-BrokerTag
Remove-BrokerApplication
Rename-BrokerApplication
Set-BrokerApplication

Parameters

-**InputObject**<Application[]>

Specifies the application to associate. Its Uid can also be substituted for the object reference.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Name**<String>

Specifies the name of the application to be associated with the desktop group.

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.712
-DesktopGroup\<DesktopGroup>\

Specifies which desktop group this application should be associated with. Note that applications can only be associated with desktop groups of the AppsOnly or DesktopsAndApps delivery type.

-Priority\<Int32>\

Specifies the priority of the mapping between the application and desktop group where lower numbers imply higher priority with zero being highest.

If one association has a higher priority than the other, machines from that group will be selected for launching sessions until all machines are at maximum load, in maintenance mode, unregistered, or unavailable for any other reason. Only when all machines from the higher-priority group are unavailable will new connections be routed to the next lowest priority group.

If multiple associations have equal priority, load balancing does not occur among the desktop groups in these associations. Instead, the broker chooses one of these groups as the preferred group and machines from this group will be selected for launching sessions until all machines are at maximum load, in maintenance mode, unregistered, or unavailable for any other reason. Only when all machines from the preferred group are unavailable will new connections be routed to another one of these groups, which the broker chooses as next-most preferred.

-LoggingId\<Guid>\

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.
-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdmin.SDK.Application, or as appropriate by property name You can pipe the application to be added to Add-BrokerApplication. You can also pipe some of the other parameters by name.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Add-BrokerApplication -BrowserName "Notepad" -DesktopGroup "Private DesktopGroup"

Adds the application with a BrowserName of "Notepad" to the desktop group called "Private DesktopGroup".
Add-BrokerDesktopGroup

Oct 10, 2013
Associate Remote PC desktop groups with the specified Remote PC catalog.

Syntax
Add-BrokerDesktopGroup [-Name] <String> [-RemotePCCatalog <Catalog>] [-Priority <Int32>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
This cmdlet forms relationships between Remote PC desktop groups and catalogs.
The Remote PC relationships are used by Remote PC automation to determine which desktop groups a machine in a particular Remote PC catalog can be published to. The assignment policy rules belonging to those desktop groups also determines the set of users that are allowed to be assigned to machines from the catalog.

Related topics
Remove-BrokerDesktopGroup
Add-BrokerCatalog
Remove-BrokerCatalog

Parameters
-InputObject <DesktopGroup[]>
Specifies one or more Remote PC desktop groups to add to a Remote PC catalog.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue) |

-Name <String>
Specifies the Remote PC desktop groups to add to a Remote PC catalog based on their name properties.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByPropertyName) |
**-RemotePCCatalog**<Catalog>

The Remote PC catalog which the desktop groups are to be added to. Specified by name, Uid or instance.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-Priority**<Int32>

Desktop group to catalog associations carry a priority number, where numerically lower values indicate a higher priority.

The priority relative to other associations determines which desktop group Remote PC automation will move a qualifying unconfigured machine into when it registers. Priority also determines which desktop group a machine will be published to when a user is assigned to the machine by Remote PC automation.

If a value is not supplied, then the desktop group association is automatically assigned a lower priority than any existing associations.

If a priority value is specified that conflicts with an existing association's priority value, then the new association is inserted with that value and existing associations are renumbered upwards to accommodate it.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>See description</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | false

Input Type
Citrix.BrokerAdmin.SDK.DesktopGroup The set of Remote PC desktop groups to be added to the catalog can be piped into this cmdlet.

Return Values
None

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerDesktopGroup -IsRemotePC $true | Add-BrokerDesktopGroup -RemotePCCatalog 42
Add all Remote PC desktop groups to Remote PC catalog 42.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Add-BrokerDesktopGroup -Name "MyGroup" -RemotePCCatalog RPCCat
Add desktop groups with names containing MyGroup to Remote PC catalog with name "RPCCat".
Add-BrokerMachine

Oct 10, 2013
Adds one or more machines to a desktop group.

Syntax
Add-BrokerMachine [-MachineName] <String> [-DesktopGroup <DesktopGroup>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
The Add-BrokerMachine cmdlet adds specified machines to a desktop group. There are three forms:

- Use the -InputObject parameter to add a single machine instance or array of instances to the group.
- Use the -MachineName parameter to add a single, named machine to the group.
- Use pipelining to pipe machines instances to the command.

The desktop group to which the machines are added can be specified by name, unique identifier (UID), or instance.

For a machine to be used in a site, the machine must be added to a desktop group. The machine and desktop group must be compatible in order for the process to succeed; for example a machine in a single-session catalog cannot be added to a multi-session desktop group.

For more information about machines, see about_Broker_Machines.

Related topics
Add-BrokerMachinesToDesktopGroup
Remove-BrokerMachine
Get-BrokerMachine

Parameters
-InputObject<Machine[]>
An array of machines to add to the group.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-MachineName<String>

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.718
The name of the single machine to add (must match the MachineName property of the machine).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-DesktopGroup

The desktop group to which the machines are added, specified by name, Uid, or instance.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdmin.SDK.Machine You can pipe in the machines you want to add.

Return Values

None

Examples
--------- EXAMPLE 1 ---------

C:\PS> Add-BrokerMachine -InputObject $machine -DesktopGroup $desktopGroup
C:\PS> Add-BrokerMachine -InputObject $machine -DesktopGroup 2
C:\PS> Add-BrokerMachine $machine -DesktopGroup "MyDesktopGroup"

These examples all add a single machine instance to a desktop group, identifying the group by instance, UID, or name.

--------- EXAMPLE 2 ---------

C:\PS> Add-BrokerMachine -MachineName "MyDomain\MyMachine" -DesktopGroup 2
C:\PS> Add-BrokerMachine "MyDomain\MyMachine" -DesktopGroup "MyDesktopGroup"
C:\PS> Add-BrokerMachine "MyDomain\MyMachine" -DesktopGroup $desktopGroup

These examples add the machine called MyMachine to a desktop group.

--------- EXAMPLE 3 ---------

C:\PS> Get-BrokerMachine -Uid 3 | Add-BrokerMachine -DesktopGroup 2
C:\PS> Get-BrokerMachine -CatalogUid 4 | Add-BrokerMachine -DesktopGroup 2

These examples find specific machines and add them to a desktop group.
Add-BrokerMachineConfiguration

Oct 10, 2013
Adds a machine configuration to a desktop group.

Syntax


Add-BrokerMachineConfiguration [-Name] <String> [-DesktopGroup <DesktopGroup>] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Associates a machine configuration with a desktop group. The settings in the machine configuration are then applied to the machines in the desktop group.

Related topics

New-BrokerMachineConfiguration
Set-BrokerMachineConfiguration
Rename-BrokerMachineConfiguration
Remove-BrokerMachineConfiguration

Parameters

-InputObject <MachineConfiguration[]>

Machine configuration to add to the desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name <String>

Name of a machine configuration to add to the desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-DesktopGroup <DesktopGroup>
The desktop group to which the machine configurations are added, specified by name, Uid, or instance.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-LoggingId**<Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress**<String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdmin.SDK.MachineConfiguration The machine configuration to add to the desktop group.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

Add-BrokerMachineConfiguration -Name UPM\Conf1 -DesktopGroup 1

Adds the machine configuration named UPM\Conf1 to the desktop group with Uid 1.

-------------------------- EXAMPLE 2 --------------------------

$mc | Add-BrokerMachineConfiguration -DesktopGroup AdminDesktops

Adds the machine configuration $mc to the desktop group named "AdminDesktops".
Add-BrokerMachinesToDesktopGroup

Oct 10, 2013

Adds machines from a catalog to a desktop group.

Syntax

Add-BrokerMachinesToDesktopGroup [-Catalog] <Catalog> [-DesktopGroup] <DesktopGroup> [-Count] <Int32> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Add-BrokerMachinesToDesktopGroup cmdlet adds a specified number of machines from a catalog to a desktop group.

The cmdlet adds as many machines as possible from the catalog to the desktop group, up to the specified number. The number of machines successfully added to the desktop group is returned.

The machines are added randomly from the catalog and are selected from those that are not already members of a desktop group, and not already assigned to a client, IP address, or user.

Both the catalog and desktop group can be referenced either by instance, name, or unique identifier (Uid). The allocation type of the catalog must be compatible with the type of desktop group. This means the session support (single/multi) and the allocation type (private/shared) of the catalog must match the session support and allocation type in the desktop group.

Related topics

Add-BrokerMachine
Remove-BrokerMachine

Parameters

- **-Catalog**<Catalog>

The catalog from which the machines are taken.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

- **-DesktopGroup**<DesktopGroup>

The desktop group to which the machines are added.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

- **-Count**<Int32>

The number of machines to add to the desktop group.
### -LoggingId<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

#### Input Type

`Citrix.BrokerAdmin.SDK.Catalog`, or the name or Uid of the catalog. You can pipe in the catalog from which the machines are taken. Alternatively, you can pipe the name or the Uid of the catalog.

#### Return Values

`System.Int32`

The number of machines added to the desktop group.

#### Examples

-------------------------- EXAMPLE 1 --------------------------

```powershell
C:\PS> Add-BrokerMachinesToDesktopGroup -Catalog $catalog -DesktopGroup $desktopGroup -Count 1000
C:\PS> Add-BrokerMachinesToDesktopGroup -Catalog "MyCatalog" -DesktopGroup "MyDesktopGroup" -Count 1000
C:\PS> Add-BrokerMachinesToDesktopGroup -Catalog 23 -DesktopGroup 4 -Count 1000
```

All these examples request that a thousand machines from a catalog are added to a desktop group. The first example references both catalog and desktop group by instance. The second example references both catalog and desktop group by name. The third example references both catalog and desktop group by Uid.

-------------------------- EXAMPLE 2 --------------------------

```powershell
C:\PS> Get-BrokerCatalog -ProvisioningType Manual | Add-BrokerMachinesToDesktopGroup -DesktopGroup $desktopGroup -Count 10
```

This example takes ten machines from each manually provisioned catalog and adds them to the specified desktop group.
Add-BrokerScope

Oct 10, 2013
Add the specified catalog/desktop group to the given scope(s).

Syntax
Add-BrokerScope [-InputObject] <Scope[]> [-Catalog <Catalog>] [-DesktopGroup <DesktopGroup>] [-LoggingId <Guid>] [-AdminAddress <String>] [<<CommonParameters>]

Detailed Description
The Add-BrokerScope cmdlet is used to associate a catalog or desktop group object with given scope(s).

To add a catalog/desktop group to a scope you need permission to change the scopes of the catalog/desktop group and permission to add objects to all of the scopes you have specified.

If the catalog/desktop group is already in any scope supplied, that scope will be silently ignored.

Related topics
Parameters
-**InputObject**<Scope[]>
  Specifies the scope(s) to add the object to. Each can take the form of either the string form of the scope's GUID or its name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Catalog**<Catalog>
  Specifies the catalog object to be added. This can take the form of an existing catalog object, a catalog Uid or name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**DesktopGroup**<DesktopGroup>
  Specifies the desktop group object to be added. This can take the form of an existing desktop group object, a desktop group Uid or name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>
Default Value

Accept Pipeline Input? true (ByValue)

-LoggingId<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Required? false
Default Value
Accept Pipeline Input? false

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

Required? false
Default Value Localhost. Once a value is provided by any cmdlet, this value will become the default.
Accept Pipeline Input? false

Input Type
Citrix.BrokerAdmin.SDK.Scope You can pipe scopes to Add-BrokerScope.

Return Values
NONE

Examples
-------------------------- EXAMPLE 1 --------------------------

C:\PS> Add-BrokerScope -Scope Chalfont –DesktopGroup 27
Adds the desktop group with Uid 27 to the Chalfont scope.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Add-BrokerScope BFC74867-C6EF-482C-996F-3E0D340E96AC -Catalog BangaloreMachines
Adds the BangaloreMachines catalog to the scope with the specified ScopeID.
Add-BrokerTag

Oct 10, 2013
Associate a tag with another object.

Syntax


Add-BrokerTag [-Name] <String> [-Desktop <Desktop>] [-DesktopGroup <DesktopGroup>] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Associates one or more tags with another object.

Related topics

Get-BrokerTag
New-BrokerTag
Remove-BrokerTag
Rename-BrokerTag

Parameters

- **InputObject**: <Tag[]>

  Specifies one or more tag objects.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name**: <String>

  Specifies a tag by name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **Desktop**: <Desktop>

  Associates the tag with a desktop.
## -DesktopGroup

Associates the tag with a desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

## -LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

## -AdminAddress

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### Input Type

Citrix.BrokerAdmin.SDK.Tag Tags may be specified through pipeline input.

### Return Values

None

### Examples

```
-------------------------- EXAMPLE 1 --------------------------
```
C:\PS> $desktop = Get-BrokerDesktop -Uid 1
C:\PS> Add-BrokerTag -Name 'Tag1' -Desktop $desktop
Associates 'Tag1' with Desktop $desktop.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> $desktop = Get-BrokerDesktop -Uid 1
C:\PS> New-BrokerTag 'Tag2' | Add-BrokerTag -Desktop $desktop
Creates a new tag with name 'Tag2' and associates it with Desktop $desktop.
Add-BrokerUser

Oct 10, 2013
Creates an association between a user and another broker object

Syntax


Detailed Description

The Add-BrokerUser cmdlet adds broker user objects to another specified object, such as a broker private desktop. This depends on the target object type:

- Machine - assign the broker machine to the specified user(s); when the machine is subsequently added to a desktop group, the desktop is also assigned to the same user(s).
- PrivateDesktop - assign the desktop to the specified user(s).
- Application - assign the application to the specified user(s).

Related topics
Get-BrokerUser
Remove-BrokerUser

Parameters

- **InputObject** <User[]>
The user objects to add.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>null</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name** <String>
The name of the user or users to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>null</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
-**Application**<Application>

The application to which the user is to be associated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Machine**<Machine>

The machine to which the user is to be assigned

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**PrivateDesktop**<PrivateDesktop>

The desktop to which the user is to be assigned

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
--- | ---
Accept Pipeline Input? | false

Input Type

Citrix.BrokerAdmin.SDK.Usr

You can pipe the users to be added to Add-BrokerUser.

Return Values

None

Notes

Specify one of the -Machine or -PrivateDesktop or -Application parameters only.

Examples

-------------------------- EXAMPLE 1 --------------------------

Add-BrokerUser "DOMAIN\UserName" -PrivateDesktop "DOMAIN\MachineName"
Assign the specified private desktop to the specified user.

-------------------------- EXAMPLE 2 --------------------------

Add-BrokerUser "DOMAIN\UserName" -Application "ApplicationName"
Assign the specified application to the specified user.
Disconnect-BrokerSession

Oct 10, 2013
Disconnect a session.

Syntax

Disconnect-BrokerSession [-InputObject] <Session[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Disconnects the specified session.

If the session is active, no warning is issued to the user before that session is disconnected.

After disconnection, sessions enter a Disconnected state. In a Disconnected state, a session still exists but there is no remote connection to that session.

Related topics

Get-BrokerSession
Stop-BrokerSession

Parameters

- **InputObject**<Session[]>

Identifies the session(s) to disconnect. This can be expressed as either a sessionUid or a session object.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host
name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.Session The sessions to disconnect can be piped into this cmdlet.

**Return Values**

None

**Notes**

This operation is non-blocking and returns before it completes. The operation, however, is unlikely to fail unless there are communication problems between the controller and the machine, or if bad arguments are passed to the cmdlet itself or if the machine cannot successfully execute the operation.

The transient nature of sessions means that the list of session objects or UIDs supplied to Disconnect-BrokerSession could consist of valid and invalid sessions. Invalid sessions are detected and disregarded and the disconnect session operation is invoked on only the valid sessions.

The system can fail to disconnect the session if the machine is not in an appropriate state or if there are problems in communicating with the machine. When a disconnect is requested the system detects if the operation was initiated successfully or not by the machine. As this operation is non-blocking the system doesn't detect or report whether the disconnect ultimately succeeded or failed after it was started.

Disconnected failures are reported through the broker SDK error handling mechanism (see about_Broker_ErrorHandling). In the event of errors the SdkErrorRecord error status is set to SessionOperationFailed and its error data dictionary is populated with the following entries:

- OperationsAttemptedCount: The number of operations attempted.
- OperationsFailedCount - The number of failed operations.
- OperationsSucceededCount - The number of successfully executed operations.
- UnresolvedSessionFailuresCount - The number of operations that failed due to invalid sessions being supplied.
- OperationInvocationFailuresCount - The number of operations that failed because they could not be invoked on the machine.
- DesktopExecutionFailuresCount - The number of operations that failed because they could not be successfully executed by the machine.

The SdkErrorRecord message will also display the number of attempted, failed and successful operations in the following format:

"Session operation error - attempted:<OperationsAttemptedCount>, failed:<OperationsFailedCount>, succeeded:
<OperationsSucceededCount>

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerSession -UserName MyDomain\MyAccount | Disconnect-BrokerSession
Attempts to disconnect all of the sessions for the user MyDomain\MyAccount.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> $desktop = Get-BrokerDesktop -DNSName MyMachine.MyDomain.com
C:\PS> Disconnect-BrokerSession $desktop.SessionUid
Disconnects the session on MyMachine.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> trap [Citrix.Broker.Admin.SDK.SdkOperationException]
C:\PS> {
C:\PS>   write ($"Exception name = " + $_.Exception.GetType().FullName)
C:\PS>   write ($"SdkOperationException.Status = " + $_.Exception.Status)
C:\PS>   write ($"SdkOperationException.ErrorData=")
C:\PS>   $_.Exception.ErrorData
C:\PS>
C:\PS>   write ($"SdkOperationException.InnerException = " + $_.Exception.InnerException)
C:\PS>   $_.Exception.InnerException
C:\PS>   continue
C:\PS> }
C:\PS> Disconnect-BrokerSession -InputObject 10,11,12
Attempts to disconnect sessions 10, 11 and 12. Traps and displays the error information.
Export-BrokerDesktopPolicy

Oct 10, 2013
Gets the site wide Citrix Group Policy settings.

Syntax

Export-BrokerDesktopPolicy [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Export-BrokerDesktopPolicy returns an array of bytes containing the site-wide Citrix Group Policy settings. These policy settings are applied to every machine in the site.

Related topics

Import-BrokerDesktopPolicy
New-BrokerConfigurationSlot
New-BrokerMachineConfiguration

Parameters

- **-AdminAddress**: <String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td></td>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

  Input Type

  None

  Return Values

  System.Byte[]

  The configuration data as an opaque binary blob. This will be null if no site wide Citrix Group Policy settings are in place.

Notes

Export-BrokerDesktopPolicy performs a specialized operation. Direct usage of it in scripts is discouraged, and could result in data corruption. It is recommended that this operation only be performed via the Citrix Studio.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> $policy = Export-BrokerDesktopPolicy
This command exports the site wide Citrix Group Policy settings.
Get-BrokerAccessPolicyRule

Oct 10, 2013
Gets rules from the site's access policy.

Syntax

Get-BrokerAccessPolicyRule [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Returns rules matching the specified search criteria from the site's access policy. If no search criteria are specified, all rules in the access policy are obtained.

An access policy rule defines a set of connection filters and access control rights relating to a desktop group. These allow fine-grained control of what access is granted to a desktop group based on details of, for example, a user's endpoint device, its address, and the user's identity.

-------------------------- BrokerAccessPolicyRule Object

A BrokerAccessPolicyRule object represents a single rule within the site's access policy. For a user to gain access to a desktop group via the rule their connection must match all its enabled include filters, and none of its enabled exclude filters. The object contains the following properties:

-- AllowedConnections (Citrix.BrokerAdmin.SDK.AllowedConnection)

Controls whether connections must be local or via Access Gateway, and if so whether specified SmartAccess tags must be provided by Access Gateway with the connection. This property forms part of the included SmartAccess tags filter.

For a detailed description of this property see "help about_Broker_AccessPolicy".

-- AllowedProtocols (System.String[]) Protocol list available to the user for sessions delivered from the rule's desktop group. If the user gains access to a desktop group by multiple rules, the allowed protocol list is the combination of the protocol lists from all those rules.

If the protocol list is empty, access to the desktop group is implicitly denied.

-- AllowedUsers (Citrix.BrokerAdmin.SDK.AllowedUser)
Controls the behavior of the included users filter. This can restrict access to a list of named users or groups, or allow access to any authenticated user. For a detailed description of this property see "help about_Broker_AccessPolicy".

--- AllowRestart (System.Boolean)
Indicates if the user can restart sessions delivered from the rule's desktop group. Session restart is handled as follows: For sessions on single-session power-managed machines, the machine is powered off, and a new session launch request made; for sessions on multi-session machines, a logoff request is issued to the session, and a new session launch request made; otherwise the property is ignored.

--- Description (System.String)
An optional description of the rule. The text is purely informational for the administrator, it is never visible to the end user.

--- DesktopGroupName (System.String)
The name of the desktop group to which the rule applies.

--- DesktopGroupUid (System.Int32)
The unique ID of the desktop group to which the rule applies.

--- Enabled (System.Boolean)
Indicates whether the rule is enabled. A disabled rule is ignored when evaluating the site's access policy.

--- ExcludedClientIPFilterEnabled (System.Boolean)
Indicates whether the excluded client IP filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

--- ExcludedClientIPs (Citrix.Broker.Admin.SDK.ChbIPAddressRange[])
IP addresses of user devices explicitly denied access to the rule's desktop group. Addresses can be specified as simple numeric addresses or as subnet masks (for example, 10.40.37.5 or 10.40.0.0/16). This property forms part of the excluded client IP address filter.

--- ExcludedClientNameFilterEnabled (System.Boolean)
Indicates whether the excluded client name filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

--- ExcludedClientNames (System.String[])
Names of user devices explicitly denied access to the rule's desktop group. This property forms part of the excluded client names filter.

--- ExcludedSmartAccessFilterEnabled (System.Boolean)
Indicates whether the excluded SmartAccess tags filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

--- ExcludedSmartAccessTags (System.String[])
SmartAccess tags which explicitly deny access to the rule's desktop group if any occur in those provided by Access Gateway with the user's connection. This property forms part of the excluded SmartAccess tags filter.
-- ExcludedUserFilterEnabled (System.Boolean)
Indicates whether the excluded users filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

-- ExcludedUsers (Citrix.BrokerAdmin.SDK.ChbUser[]) 
Users and groups who are explicitly denied access to the rule's desktop group. This property forms part of the excluded users filter.

-- HdxSslEnabled (System.Boolean)
Indicates whether SSL encryption is enabled for sessions delivered from the rule's desktop group.

-- IncludedClientIPFilterEnabled (System.Boolean)
Indicates whether the included client IP filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

-- IncludedClientIPs (Citrix.BrokerAdmin.SDK.ChbIPAddressRange[])
IP addresses of user devices allowed access to the rule's desktop group. Addresses can be specified as simple numeric addresses or as subnet masks (for example, 10.40.37.5 or 10.40.0.0/16). This property forms part of the included client IP address filter.

-- IncludedClientNameFilterEnabled (System.Boolean)
Indicates whether the included client names filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

-- IncludedClientNames (System.String[])
Names of user devices allowed access to the rule's desktop group. This property forms part of the included client names filter.

-- IncludedSmartAccessFilterEnabled (System.Boolean)
Indicates whether the included SmartAccess tags filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

-- IncludedSmartAccessTags (System.String[])
The SmartAccess tags which grant access to the rule's desktop group if any occur in those provided by Access Gateway with the user's connection. This property forms part of the excluded SmartAccess tags filter.

-- IncludedUserFilterEnabled (System.Boolean)
Indicates whether the included users filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

-- IncludedUsers (Citrix.BrokerAdmin.SDK.ChbUser[])
Users and groups who are granted access to the rule's desktop group. This property forms part of the included users filter.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
A collection of arbitrary key/value pairs that can be associated with the rule. The administrator can use these values for any purpose; they are not used by the site itself in any way.
-- Name (System.String)

Administrative name of the rule. Each rule in the site's access policy must have a unique name.

-- Uid (System.Int32)

Unique ID of the rule itself.

Related topics

New-BrokerAccessPolicyRule
Set-BrokerAccessPolicyRule
Rename-BrokerAccessPolicyRule
Remove-BrokerAccessPolicyRule

Parameters

-Uid<Int32>

Gets only the rule with the specified unique ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Name<String>

Gets only rules with the specified name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AllowedConnections<AllowedConnection>

Gets only rules that have the specified value in the AllowedConnections property of their included SmartAccess tags filter.

Valid values are Filtered, NotViaAG, and ViaAG.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AllowedUsers<AllowedUser>
Gets only rules that have the specified value in the AllowedUsers property of their included users filter.

Valid values are Filtered, AnyAuthenticated, and Any.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowedUsers</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-Description<If using the -Description parameter, it gets only rules with the specified description.>

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-DesktopGroupName<If using the -DesktopGroupName parameter, it gets only rules applying to desktop groups with names matching the specified name.>

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>DesktopGroupName</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-DesktopGroupUid<If using the -DesktopGroupUid parameter, it gets only rules that apply to the desktop group with the specified unique ID.>

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>DesktopGroupUid</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-Enabled<If using the -Enabled parameter, it gets only rules that are in the specified state, either enabled ($true) or disabled ($false).>

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enabled</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
**-ExcludedClientIPFilterEnabled**<Boolean>

Gets only rules that have their excluded client IP address filter enabled ($true) or disabled ($false).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ExcludedClientName**<String>

Gets only rules that have the specified client name in their excluded client names filter (whether the filter is enabled or not).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ExcludedClientNameFilterEnabled**<Boolean>

Gets only rules that have their excluded client name filter enabled ($true) or disabled ($false).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ExcludedSmartAccessFilterEnabled**<Boolean>

Gets only rules that have their excluded SmartAccess tags filter enabled ($true) or disabled ($false).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ExcludedSmartAccessTag**<String>

Gets only rules that have the specified SmartAccess tag in their excluded SmartAccess tags filter (whether the filter is enabled or not).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
### -ExcludedUser

Gets only rules that have the specified user in their excluded users filter (whether the filter is enabled or not).

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -ExcludedUserFilterEnabled

Gets only rules that have their excluded user filter enabled ($true) or disabled ($false).

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -IncludedClientIPFilterEnabled

Gets only rules that have their included client IP address filter enabled ($true) or disabled ($false).

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -IncludedClientName

Gets only rules that have the specified user device name in their included client names filter (whether the filter is enabled or not).

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -IncludedClientNameFilterEnabled

Gets only rules that have their included client name filter enabled ($true) or disabled ($false).

| Required? | false |
| Default Value | |
Accept Pipeline Input? | false

**-IncludedSmartAccessFilterEnabled**<boolean>

Gets only rules that have their included SmartAccess tags filter enabled (true) or disabled (false).

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-IncludedSmartAccessTag**<string>

Gets only rules that have the specified SmartAccess tag in their included SmartAccess tags filter (whether the filter is enabled or not).

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-IncludedUser**<User>

Gets only rules that have the specified user in their included users filter (whether the filter is enabled or not).

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-IncludedUserFilterEnabled**<boolean>

Gets only rules that have their included user filter enabled (true) or disabled (false).

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-Metadata**<string>

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".
<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>-ReturnTotalRecordCount</strong></td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td><strong>-MaxRecordCount</strong></td>
<td>false</td>
<td>250</td>
<td>false</td>
</tr>
<tr>
<td><strong>-Skip</strong></td>
<td>false</td>
<td>0</td>
<td>false</td>
</tr>
<tr>
<td><strong>-SortBy</strong></td>
<td>false</td>
<td>The default sort order is by name or unique identifier.</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-ReturnTotalRecordCount**<SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

- **-MaxRecordCount**<Int32>

Specifies the maximum number of records to return.

- **-Skip**<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

- **-SortBy**<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.
-Filter&lt;String&gt;

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-Property&lt;String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AdminAddress&lt;String&gt;

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

Input Type
None You cannot pipe input into this cmdlet.

Return Values
Citrix.BrokerAdmin.SDK.AccessPolicyRule

Get-BrokerAccessPolicyRule returns all access policy rules that match the specified selection criteria.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerAccessPolicyRule
Returns all access policy rules. This offers a complete description of the current site's access policy.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-BrokerAccessPolicyRule -Enabled $true -IncludedUser sales\tech-support
Returns all rules that are both enabled and explicitly include the SALES\tech-support group in their included users filter.
Get-BrokerAppAssignmentPolicyRule

Oct 10, 2013
Gets application rules from the site's assignment policy.

Syntax

Get-BrokerAppAssignmentPolicyRule [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Returns application rules matching the specified search criteria from the site's assignment policy. If no search criteria are specified, all application rules in the assignment policy are obtained.

An application rule in the assignment policy defines the users who are entitled to a self-service persistent machine assignment from the rule's desktop group; once assigned the machine can run one or more applications published from the group.

-------------------------- BrokerAppAssignmentPolicyRule Object

The BrokerAppAssignmentPolicyRule object represents a single application rule within the site's assignment policy. It contains the following properties:

-- Description (System.String)
An optional description of the rule. The text is purely informational for the administrator, it is never visible to the end user.

-- DesktopGroupUid (System.Int32)
The unique ID of the desktop group to which the rule applies.

-- Enabled (System.Boolean)
Indicates whether the rule is enabled. A disabled rule is ignored when evaluating the site's assignment policy.

-- ExcludedUserFilterEnabled (System.Boolean)
Indicates whether the excluded users filter is enabled. If the filter is disabled then any user entries in the filter are ignored when assignment policy rules are evaluated.

-- ExcludedUsers (Citrix.BrokerAdmin.SDK.ChbUser[]) The excluded users filter of the rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from the rule's desktop group.

-- IncludedUserFilterEnabled (System.Boolean)
Indicates whether the included users filter is enabled. If the filter is disabled then any user who satisfies the requirements of
the access policy is implicitly entitled to the machine assignment described by the rule.

-- IncludedUsers (Citrix.BrokerAdminSDK.ChbUser[])  
The included users filter of the rule, that is, the users and groups who are entitled to a machine assignment from the rule's desktop group.

-- Name (System.String)  
The administrative name of the rule. Each rule in the site's assignment policy must have a unique name (irrespective of whether they are desktop or application rules).

-- Uid (System.Int32)  
The unique ID of the rule itself.

Related topics

New-BrokerAppAssignmentPolicyRule
Set-BrokerAppAssignmentPolicyRule
Rename-BrokerAppAssignmentPolicyRule
Remove-BrokerAppAssignmentPolicyRule

Parameters

-Uid<Int32>  
Gets the application rule with the specified unique ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Name<String>  
Gets only application rules with the specified name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Description<String>  
Gets only application rules with the specified description.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>Accept Pipeline Input?</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------</td>
</tr>
<tr>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-DesktopGroupUid**<Int32>

  Gets only application rules that apply to the desktop group with the specified unique ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Enabled**<Boolean>

  Gets only application rules that are in the specified state, either enabled ($true) or disabled ($false).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-ExcludedUser**<User>

  Gets only application rules that have the specified user in their excluded users filter (whether the filter is enabled or not)

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-ExcludedUserFilterEnabled**<Boolean>

  Gets only application rules that have their excluded user filter enabled ($true) or disabled ($false).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-IncludedUser**<User>

  Gets only application rules that have the specified user in their included users filter (whether the filter is enabled or not).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
### -IncludedUserFilterEnabled <Boolean>

Gets only application rules that have their included user filter enabled ($true) or disabled ($false).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

### -ReturnTotalRecordCount <SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>False</td>
<td>false</td>
</tr>
</tbody>
</table>

### -MaxRecordCount <Int32>

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>250</td>
<td>false</td>
</tr>
</tbody>
</table>

### -Skip <Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>0</td>
<td>false</td>
</tr>
</tbody>
</table>

### -SortBy <String>
Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Filter**<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

None You cannot pipe input into this cmdlet.

**Return Values**

Citrix.BrokerAdmin.SDK.AppAssignmentPolicyRule
Get-BrokerAppAssignmentPolicyRule returns all application rules in the assignment policy that match the specified selection criteria.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerAppAssignmentPolicyRule
Returns all application rules from the assignment policy. This offers a complete description of the current site's assignment policy with respect to machine assignment entitlements for delivery of application sessions from private desktop groups.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> $dg = Get-BrokerDesktopGroup 'Sales Support'
C:\PS> Get-BrokerAppAssignmentPolicyRule -DesktopGroupUid $dg.Uid
Returns the rule in the assignment policy that gives users entitlements to machine assignments in the Sales Support desktop group for delivery of application sessions.
Get-BrokerAppEntitlementPolicyRule

Oct 10, 2013
Gets application rules from the site's entitlement policy.

Syntax

Get-BrokerAppEntitlementPolicyRule [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Returns application rules matching the specified search criteria from the site's entitlement policy. If no search criteria are specified, all application rules in the entitlement policy are obtained.

An application rule in the entitlement policy defines the users who are allowed per-session access to a machine to run one or more applications published from the rule's desktop group.

-------------------------- BrokerAppEntitlementPolicyRule Object

The BrokerAppEntitlementPolicyRule object represents a single application rule within the site's entitlement policy. It contains the following properties:

-- Description (System.String)
Optional description of the rule. The text is purely informational for the administrator; it is never visible to the end user.

-- DesktopGroupUid (System.Int32)
The unique ID of the desktop group to which the rule applies.

-- Enabled (System.Boolean)
Indicates whether the rule is enabled. A disabled rule is ignored when evaluating the site's entitlement policy.

-- ExcludedUserFilterEnabled (System.Boolean)
Indicates whether the excluded users filter of the rule is enabled. If the filter is disabled then any user entries in the filter are ignored when entitlement policy rules are evaluated.

-- ExcludedUsers (Citrix.BrokerAdmin.SDK.ChbUser[])
The excluded users filter of the rule, that is, the users and groups who are explicitly denied entitlements to use published applications from the associated desktop group.

-- IncludedUserFilterEnabled (System.Boolean)
Indicates whether the included users filter of the rule is enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to an application session by the rule.
-- IncludedUsers (Citrix.Broker.Admin.SDK.ChbUser[])  
The included users filter of the rule, that is, the users and groups who are granted an entitlement to an application session
by the rule.

If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded
users filter, no entitlement is granted whether or not the user appears in the included users filter.

-- Name (System.String)  
The administrative name of the rule. Each rule in the site's entitlement policy must have a unique name (irrespective of
whether they are desktop or application rules).

-- Uid (System.Int32)  
The unique ID of the rule itself.

Related topics

New-BrokerAppEntitlementPolicyRule
Set-BrokerAppEntitlementPolicyRule
Rename-BrokerAppEntitlementPolicyRule
Remove-BrokerAppEntitlementPolicyRule

Parameters

- **Uid**<Int32>  
Gets the application rule with the specified unique ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Name**<String>  
Gets only application rules with the specified name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Description**<String>  
Gets only application rules with the specified description.
-DesktopGroupUid<int32>

Gets only the application rule that applies to the desktop group with the specified unique ID.

-Enabled<boolean>

Gets only application rules that are in the specified state, either enabled (true), or disabled (false).

-_ExcludedUser<User>

Gets only application rules that have the specified user in their excluded users filter (whether the filter is enabled or not).

-ExcludedUserFilterEnabled<boolean>

Gets only application rules that have their excluded user filter enabled (true) or disabled (false).

-IncludedUser<User>

Gets only application rules that have the specified user in their included users filter (whether the filter is enabled or not).
### IncludedUserFilterEnabled <Boolean>

Gets only application rules that have their included user filter enabled (true) or disabled (false).

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### ReturnTotalRecordCount <SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

### MaxRecordCount <Int32>

Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |
| Accept Pipeline Input? | false |

### Skip <Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | false |
-**SortBy**<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Filter**<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values
Get-BrokerAppEntitlementPolicyRule returns all application rules from the entitlement policy that match the specified selection criteria.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerAppEntitlementPolicyRule
Returns all application rules from the entitlement policy. This offers a complete description of the current site's entitlement policy with respect to application entitlements from shared desktop groups.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> $dg = Get-BrokerDesktopGroup 'Customer Support'
C:\PS> Get-BrokerAppEntitlementPolicyRule -DesktopGroupUid $dg.Uid
Returns the application rule in the entitlement policy that grants users an entitlement to application sessions in the Customer Support desktop group.
Get-BrokerApplication

Oct 10, 2013
Get the applications published on this site.

Syntax

Get-BrokerApplication [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

The Get-BrokerApplication cmdlet gets the published applications that are hosted on this site.

Without parameters, Get-BrokerApplication gets all the applications that have been published, regardless of whether they are visible to users or not. You can also use the parameters of Get-BrokerApplication to filter the results to just the applications you're interested in. You can also identify applications by their UIDs or their BrowserNames.

For more information about applications, see about_Broker_Applications.

-------------------------- BrokerApplication Object

The BrokerApplication object represents a published application in the site. It contains the following properties:

-- ApplicationType (Citrix.BrokerAdmin.SDK.ApplicationType)

The type of the application, whether HostedOnDesktop or InstalledOnClient.

-- AssociatedDesktopGroupPriorities (SystemInt32[])

List of associated desktop group priorities. Associated desktop groups is the list of desktop groups on which the application is published. When launching an application an available machine from one of the associated groups is selected. Desktop groups are searched for available machines in order of their priority.

-- AssociatedDesktopGroupUids (SystemInt32[])

List of associated desktop group uids. Associated desktop groups is the list of desktop groups on which the application is published. The list is sorted by priority, with the highest priority group first.

-- AssociatedUserFullNames (SystemString[])

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
List of associated users (full names). Associated users is the list of users who are given access using the application/user mapping filter.

-- AssociatedUserNames (System.String[])  
List of associated users (SAM names). Associated users is the list of users who are given access using the application/user mapping filter.

-- AssociatedUserUPNs (System.String[])  
List of associated users (user principle names). Associated users is the list of users who are given access using the application/user mapping filter.

-- BrowserName (System.String)  
Unique browser name used to identify this application to other components in the site. This value is not visible to the end users.

-- ClientFolder (System.String)  
The folder that the application belongs to as the user sees it.

-- CommandLineArguments (System.String)  
The command-line arguments to use when launching the executable.

-- CommandLineExecutable (System.String)  
The name including the full path of the executable file to launch.

-- CpuPriorityLevel (Citrix.BrokerAdmin.SDK.CpuPriorityLevel)  
The CPU priority of the launched process. Valid values are: Low, BelowNormal, Normal, AboveNormal, and High.

-- Description (System.String)  
Optional application description. This description is visible to the end users.

-- Enabled (System.Boolean)  
Specifies whether or not this application can be launched.

-- IconFromClient (System.Boolean)  
Specifies if the app icon should be retrieved from the application on the client. This is reserved for possible future use, and all applications of type HostedOnDesktop cannot set or change this value.

-- IconUid (System.Int32?)  
The icon UID used for this application. If not specified a generic icon is used.

-- MetadataKeys (System.String[])  
All key names of metadata items associated with this application.
MetadataMap (System.Collections.Generic.Dictionary<string, string>)

Metadata for this application.

-- Name (System.String)

Unique administrative name of application.

-- PublishedName (System.String)

Published name of application as seen by end user. If not specified value used defaults to the administrative name.

-- SecureCmdLineArgumentsEnabled (System.Boolean)

Specifies whether the command-line arguments should be secured.

-- ShortcutAddedToDesktop (System.Boolean)

Specifies whether a shortcut to the application should be placed on the user device.

-- ShortcutAddedToStartMenu (System.Boolean)

Specifies whether a shortcut to the application should be placed in the user's Start menu on their user device.

-- StartMenuFolder (System.String)

The name of the Start menu folder that holds the application shortcut.

-- Uid (System.Int32)

A unique identifier of an application.

-- UserFilterEnabled (System.Boolean)

Indicates if application-specific user filter is enabled.

-- UUID (System.Guid)

UUID of the application.

-- Visible (System.Boolean)

Specifies if the application is visible to users.

-- WaitForPrinterCreation (System.Boolean)

Specifies whether the VDA delays starting the app until printers are set up or not.

-- WorkingDirectory (System.String)

The working directory the executable is launched from.

Related topics

New-BrokerApplication
Add-BrokerApplication
Remove-BrokerApplication
Rename-BrokerApplication
Set-BrokerApplication

Parameters

- **-Uid]**<Int32>
  Gets only the application with the specified unique identifier.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | false |

- **-Name]**<String>
  Gets only the applications matching the specified name.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

- **-ApplicationType]**<ApplicationType>
  Gets applications that match the type specified: HostedOnDesktop or InstalledOnClient.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

- **-AssociatedDesktopGroupPriority]**<Int32>
  Gets applications with an associated desktop group identified by priority assigned to the pairing between an application and desktop group.
  
  Associated desktop group is a desktop group on which the application is published.

<p>| Required? | false |
| Default Value |</p>
<table>
<thead>
<tr>
<th><strong>Accept Pipeline Input?</strong></th>
<th>false</th>
</tr>
</thead>
</table>

- **AssociatedDesktopGroupUid**<Int32>

Gets applications with an associated desktop group identified by the desktop group UID.

Associated desktop group is a desktop group on which the application is published.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AssociatedUserFullName**<String>

Gets applications with an associated user identified by their full name (usually 'first-name last-name').

If the 'UserFilterEnabled' property is true then access to the application is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AssociatedUserName**<String>

Gets applications with an associated user identified by their user name (in the form 'domain\user'). If the 'UserFilterEnabled' property is true then access to the application is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AssociatedUserUPN**<String>

Gets applications with an associated user identified by their user principle name (in the form 'user@domain'). If the 'UserFilterEnabled' property is true then access to the application is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
- **BrowserName** <String>

  Gets only the applications that match the supplied name. The BrowserName is usually an internal name for the application and is unique in the site.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ClientFolder** <String>

  Gets only the applications that match the specified value for the folder the application belongs to as seen by the end-user. This folder can be seen in the Citrix Online Plug-in, in Web Services, and also potentially in the user's start menu.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **CommandLineArguments** <String>

  Gets only the applications that match the supplied arguments to the command-line executable.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **CommandLineExecutable** <String>

  Gets only the applications that match the supplied command-line executable.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **CpuPriorityLevel** <CpuPriorityLevel>

  Gets only the applications that have the specified value for the CPU priority level of the launched executable. Valid values are: Low, BelowNormal, Normal, AboveNormal, and High.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
-Description <String>

Gets only the applications that match the supplied description.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Enabled <Boolean>

Gets only the applications that have the specified value for whether the application is enabled. Disabled applications are still visible to users (that is controlled by the Visible setting) but cannot be launched.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-IconFromClient <Boolean>

Gets only the applications that have the specified value for whether the application icon should be retrieved from the user device.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-IconUid <Int32>

Gets only the applications that use the specified icon (identified by its Uid).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-MetadataKey <String>

Gets only applications whose associated metadata contains key names matching the specified value.
<table>
<thead>
<tr>
<th><strong>-Metadata</strong>&lt;String&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets records with matching metadata entries.</td>
<td></td>
</tr>
<tr>
<td>The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata &quot;abc:x*&quot; matches records with a metadata entry having a key name of &quot;abc&quot; and a value starting with the letter &quot;x&quot;.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-PublishedName</strong>&lt;String&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets applications whose published name matches the supplied pattern.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-SecureCmdLineArgumentsEnabled</strong>&lt;Boolean&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets only the applications that have the specified value for whether the command-line arguments should be secured. This is reserved for possible future use, and all applications of type HostedOnDesktop can only have this value set to true.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-ShortcutAddedToDesktop</strong>&lt;Boolean&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets only the applications that match depending on whether a shortcut for the application has been added to the user device or not.</td>
<td></td>
</tr>
</tbody>
</table>
-**ShortcutAddedToStartMenu**<Boolean>

Gets only the applications that match depending on whether a shortcut for the application has been added to Start Menu of the user device or not.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**StartMenuFolder**<String>

Gets only the applications that match the specified name for the start menu folder that holds the application shortcut. This is valid only for the Citrix Online Plug-in.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**UserFilterEnabled**<Boolean>

Gets only applications whose user filter is in the specified state.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**UUID**<Guid>

Gets applications with the specified value of UUID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Visible**<Boolean>

Gets only the applications that have the specified value for whether it is visible to the users.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Accept Pipeline Input?</strong></th>
<th><code>false</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>-WaitForPrinterCreation</strong>&lt;Boolean&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td><code>false</code></td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><code>false</code></td>
</tr>
<tr>
<td><strong>-WorkingDirectory</strong>&lt;String&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td><code>false</code></td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><code>false</code></td>
</tr>
<tr>
<td><strong>-DesktopUid</strong>&lt;Int32&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td><code>false</code></td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><code>false</code></td>
</tr>
<tr>
<td><strong>-SessionUid</strong>&lt;Int64&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td><code>false</code></td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><code>false</code></td>
</tr>
<tr>
<td><strong>-UserSID</strong>&lt;String&gt;</td>
<td></td>
</tr>
</tbody>
</table>
### -DesktopGroupUid

**Int32**

Gets only the applications that have been published to the specified desktop group (identified by its Uid).

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -ReturnTotalRecordCount

**SwitchParameter**

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

### -MaxRecordCount

**Int32**

Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |
| Accept Pipeline Input? | false |

### -Skip

**Int32**

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | false |

### -SortBy

**String**
Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |
| Accept Pipeline Input? | false |

- **Filter**<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

- **Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.Application
Get-BrokerApplication returns an object for each application it gets.

Notes

Get-BrokerApplication returns just the application object, and as such is not a complete picture. The returned objects do not tell you what File-Type Associations are configured for this application, etc.

Use the following cmdlets to gather data related to applications (shown with examples of syntax):

- Get-BrokerConfiguredFTA -ApplicationUid $app.Uid
- Get-BrokerTag -ApplicationUid $app.Uid
- Get-BrokerDesktopGroup -ApplicationUid $app.Uid
- Get-BrokerDesktop -PublishedApplication $app
- Get-BrokerSession -ApplicationUid $app.Uid
- Get-BrokerApplicationInstance -ApplicationUid $app.Uid

Examples

-------------------------- EXAMPLE 1--------------------------

C:\PS> Get-BrokerApplication Notepad
Returns the application with the Name of "Notepad".

-------------------------- EXAMPLE 2--------------------------

C:\PS> Get-BrokerApplication -PublishedName Note* -Enabled $true
Returns the applications that have a PublishedName starting with "Note" and that are enabled.
Get-BrokerApplicationInstance

Oct 10, 2013
Gets the running applications on the desktops.

Syntax

Get-BrokerApplicationInstance -Uid <Int64> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

The Get-BrokerApplicationInstance gets the published applications that are running on desktops.

Only published applications that are launched from a Citrix client are returned. If a user launches an application from within a session (by double-clicking on an attachment from an email, for example) this will not show up in the list of running applications.

Also note that this is a list of launched published applications, not a list of processes running on the desktop. In some cases the original process associated with the published application may no longer be running, but if the session is still running the published application may be listed as running.

The number of instances for each published application running in a session is also returned. For example, if a user launches two Notepad applications from a Citrix client, and session-sharing occurs such that both Notepad applications run in the same session, then the Instances property indicates that 2 copies are running in the session.

See Get-BrokerApplication and Get-BrokerSession to get the details for the applications and sessions, respectively.

The Get-BrokerMachine cmdlet also returns a list of published applications that are running on a desktop. See the "ApplicationsInUse" attribute of the returned desktop objects.

-------------------------- BrokerApplicationInstance Object

The BrokerApplicationInstance object represents an instance of a published application in the site. It contains the following properties:

-- ApplicationName (System.String)

The administrative name of the application.

-- ApplicationUid (System.Int32)

The UID of the application.

-- Instances (System.Int32)

The number of times this published application is running in the specified session.
-- MachineName (System.String)
Machine's SAM name (of the form domain\machine). If SAM name is unavailable, contains the machines's SID.

-- MachineUid (System.Int32)
UID of underlying machine.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
Metadata for this application instance.

-- SessionUid (System.Int64)
The UID of the session.

-- Uid (System.Int64)
The unique identifier for this application instance object itself, distinct from the Uids of either application or session objects.

-- UserName (System.String)
User name (SAMName).

Related topics
Get-BrokerApplication
Get-BrokerDesktop
Get-BrokerSession

Parameters
-Uid<Int64>
Gets only the application instances specified by the unique identifier. This is the unique identifier for the application instance object itself, and is distinct from the Uids of either application or session objects.

<table>
<thead>
<tr>
<th>Required?</th>
<th>True</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

-ApplicationName<String>
Gets only application instances for the specified application name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------</td>
</tr>
</tbody>
</table>

**-ApplicationUid<Int32>**

Gets only application instances for the specified application Uid.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Instances<Int32>**

Gets only application instances that match the specified number of instances.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-MachineName<String>**

Gets only application instances running on the specified machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-MachineUid<Int32>**

Gets only application instances running on the machine with the specified UID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Metadata<String>**

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
-**SessionUid**<Int64>

Gets only application instances for the published applications running in the specified session. Note, you must specify the SessionUid, not the SessionID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**UserName**<String>

Gets only application instances being run by the specified users.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**ReturnTotalRecordCount**<SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**MaxRecordCount**<Int32>

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SortBy<String>**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter<String>**

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Property<String[]>**

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
---|---
Accept Pipeline Input? | false

Input Type
None You cannot pipe input into this cmdlet.

Return Values
Citrix.BrokerAdmin.SDK.ApplicationInstance
Get-BrokerApplicationInstance returns an object for each application instance it gets.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerApplicationInstance -Uid 3
Returns the application instance with a Uid of 3. Note that this is the unique identifier for application instances, which is distinct from the unique identifiers of either application or session objects.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> $app = Get-BrokerApplication Notepad
C:\PS> Get-BrokerApplicationInstance -ApplicationUid $app.Uid
Returns all the application instances for the Notepad application. Use this to see if there are any launched instances of Notepad running in your site and, if so, from which desktops.

-------------------------- EXAMPLE 3 --------------------------
C:\PS> $sessions = Get-BrokerSession -MachineName "ACME\Worker1"
C:\PS> for ($i=0; $i -lt $sessions.Length; $i++) {
        Get-BrokerApplicationInstance -SessionUid $sessions[$i].SessionUid
    }
Returns all the applications that are running on the "Worker1" machine in the "ACME" domain. Use this to see which published applications are running on a specific machine.

Note that the SessionUid, not the SessionId, is specified as a parameter to this cmdlet. The SessionId is a unique identifier that Remote Desktop Services uses to track the session, and is unique only on that machine. The SessionUid, on the other hand, is unique across the entire site.

The "ApplicationsInUse" attribute of the returned session object also provides a list of running launched applications, and in many cases might be more convenient to use. It returns a list of application BrowserNames.
Get-BrokerAssignmentPolicyRule

Oct 10, 2013
Gets desktop rules from the site's assignment policy.

Syntax

Get-BrokerAssignmentPolicyRule [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Returns desktop rules matching the specified search criteria from the site's assignment policy. If no search criteria are specified, all desktop rules in the assignment policy are obtained.

A desktop rule in the assignment policy defines the users who are entitled to self-service persistent machine assignments from the rule's desktop group. A rule defines how many machines a user is allowed from the group for delivery of full desktop sessions.

-------------------------- BrokerAssignmentPolicyRule Object

The BrokerAssignmentPolicyRule object represents a single desktop rule within the site's assignment policy. It contains the following properties:

-- ColorDepth (Citrix.BrokerAdmin.SDK.ColorDepth?)

The color depth of desktop sessions launched by the user from machines assigned to them by the rule. If null, the equivalent setting from the rule's desktop group is used.

-- Description (System.String)

Optional description of the rule. The text may be visible to the end user, for example, as a tooltip associated with the desktop entitlement.

-- DesktopGroupUid (System.Int32)

The unique ID of the desktop group to which the rule applies.

-- Enabled (System.Boolean)

Indicates whether the rule is enabled. A disabled rule is ignored when evaluating the site's assignment policy.

-- ExcludedUserFilterEnabled (System.Boolean)

Indicates whether the excluded users filter is enabled. If the filter is disabled then any user entries in the filter are ignored when assignment policy rules are evaluated.
The excluded users filter of the rule, that is, the users and groups who are explicitly denied entitlements to machine assignments from the rule's desktop group.

The unique ID of the icon used for the machine entitlement seen by the user or, after a machine is assigned by the rule, the icon for the desktop itself. If null, the equivalent setting from the rule's desktop group is used.

Indicates whether the included users filter is enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly entitled to the machine assignments described by the rule.

For rules that relate to RemotePC desktop groups however, if the included user filter is disabled, the rule is effectively disabled.

The included users filter of the rule, that is, the users and groups who are entitled to machine assignments from the rule's desktop group.

The number of machines from the rule's desktop group to which a user is entitled. Where an entitlement is granted to a user group rather than an individual, the number of machines applies to each member of the user group independently.

A collection of arbitrary key/value pairs that can be associated with the rule. The administrator can use these values for any purpose; they are not used by the site itself in any way.

The administrative name of the rule. Each rule in the site's assignment policy must have a unique name (irrespective of whether they are desktop or application rules).

The published name of the desktop entitlement seen by the user or, after a machine is assigned by the rule, the published name of the desktop itself. If null, the equivalent setting from the rule's desktop group is used.

Indicates whether the rule requires the SecureICA protocol for desktop sessions launched using a machine assigned by the rule. If null, the equivalent setting from the rule's desktop group is used.

The unique ID of the rule itself.

The unique ID of the rule itself.
UUID of the rule.

Related topics
New-BrokerAssignmentPolicyRule
Set-BrokerAssignmentPolicyRule
Rename-BrokerAssignmentPolicyRule
Remove-BrokerAssignmentPolicyRule

Parameters
-Uid<Int32>
Gets the desktop rule with the specified unique ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>true</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Name<String>
Gets only desktop rules with the specified name.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ColorDepth<ColorDepth>
Gets only desktop rules with the specified color depth.
Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Description<String>
Gets only desktop rules with the specified description.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>
### Default Value

| Accept Pipeline Input? | false |

<table>
<thead>
<tr>
<th><strong>-DesktopGroupUid</strong>&lt;Int32&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets only desktop rules that apply to the desktop group with the specified unique ID.</td>
</tr>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-Enabled</strong>&lt;Boolean&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets only rules that are in the specified state, either enabled ($true) or disabled ($false).</td>
</tr>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-ExcludedUser</strong>&lt;User&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets only desktop rules that have the specified user in their excluded users filter (whether the filter is enabled or not).</td>
</tr>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-ExcludedUserFilterEnabled</strong>&lt;Boolean&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets only desktop rules that have their excluded user filter enabled ($true) or disabled ($false).</td>
</tr>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-IconUid</strong>&lt;Int32&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets only desktop rules using the icon with the specified unique ID.</td>
</tr>
<tr>
<td>Required?</td>
</tr>
</tbody>
</table>
### -IncludedUser\(<\text{User}>\)

Gets only desktop rules that have the specified user in their included users filter (whether the filter is enabled or not).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

### -IncludedUserFilterEnabled\(<\text{Boolean}>\)

Gets only desktop rules that have their included user filter enabled (true) or disabled (false).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

### -MaxDesktops\(<\text{Int32}>\)

Gets only desktop rules granting the specified number of machine assignment entitlements.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

### -Metadata\(<\text{String}>\)

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

### -PublishedName\(<\text{String}>\)

Gets only desktop rules with the specified published name, that is, the desktop name that the end user sees.
**-SecureIcaRequired**<Boolean>

Gets only desktop rules that require desktop sessions to machines assigned by the rule to use the SecureICA protocol ($true) or not ($false).

**-UUID**<Guid>

Gets rules with the specified value of UUID.

**-ReturnTotalRecordCount**<SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

**-MaxRecordCount**<Int32>

Specifies the maximum number of records to return.
-Skip<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-SortBy<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Filter<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Property<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.
Required? false

Default Value Localhost. Once a value is provided by any cmdlet, this value will become the default.

Accept Pipeline Input? false

Input Type None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.AssignmentPolicyRule

Get-BrokerAssignmentPolicyRule returns all assignment policy rules that match the specified selection criteria.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerAssignmentPolicyRule
Returns all desktop rules from the assignment policy. This offers a complete description of the current site’s assignment policy with respect to machine assignment entitlements for delivery of desktop sessions from private desktop groups.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> $dg = Get-BrokerDesktopGroup 'Sales Support'
C:\PS> Get-BrokerAssignmentPolicyRule -DesktopGroupUid $dg.Uid
Returns all rules in the assignment policy that give users entitlements to machine assignments in the Sales Support desktop group for delivery of full desktop sessions.
Get-BrokerCatalog

Oct 10, 2013
Gets catalogs configured for this site.

Syntax

Get-BrokerCatalog [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Retrieves catalogs matching the specified criteria. If no parameters are specified this cmdlet enumerates all catalogs.

See about_Broker_Filtering for information about advanced filtering options.

------------------------- BrokerCatalog Object

The catalog object returned represents a group of related physical or virtual machines that have been configured in the site.

See about_Broker_Machines for more information.

-- AllocationType (Citrix.Broker.Admin.SDK.AllocationType)

Denotes how the the machines in the catalog are allocated to a user.

Possible values are:

- Static: Machines get assigned to a user either by the admin or on first use. This relationship is static and changes only if an admin explicitly changes the assignments.
- Random: Machines are allocated to users randomly from a pool of available machines.

-- AssignedCount (System.Int32)

The number of assigned machines (machines that have been assigned to a user/users or a client name/address).

-- AvailableAssignedCount (System.Int32)

The number of available machines (not in a desktop group), that are also assigned to users.

-- AvailableCount (System.Int32)

The number of available machines (those not in any desktop group).
-- AvailableUnassignedCount (System.Int32)

The number of available machines (those not in any desktop group) that are not assigned to users.

-- Description (System.String)

Description of the catalog.

-- HypervisorConnectionUid (System.Int32?)

The Uid of the hypervisor connection that is associated with the machines in the catalog. This property only applies to MCS provisioned catalogs. For other provisioning types machines can be from one or more different hypervisor connections.

-- IsRemotePC (System.Boolean)

Specifies whether or not the catalog is a RemotePC catalog. Remote PC catalogs automatically configure appropriate machines without the need for manual configuration. See about_Broker_RemotePC for more information.

-- MachinesArePhysical (System.Boolean)

Specifies whether or not the machines in the catalog can be power-managed by the broker.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

Holds any metadata associated with the catalog.

-- MinimumFunctionalLevel (Citrix.BrokerAdmin.SDK.FunctionalLevel)

The expected minimal functional level of the machines in the catalog.

-- Name (System.String)

Name of the catalog.

-- PersistUserChanges (Citrix.BrokerAdmin.SDKPersistUserChanges)

Specifies how the user changes are persisted on the machines in the catalog. Possible values are:

- OnLocal: The user changes are stored on the machine's local storage.
- Discard: The user changes are discarded.
- OnPvd: The user changes are stored on the user's personal vDisk.

-- ProvisioningSchemeId (System.Guid?)

The GUID of the provisioning scheme (if any) associated with the catalog. This only applies if the provisioning type is MCS.

-- ProvisioningType (Citrix.BrokerAdmin.SDK.ProvisioningType)

Specifies how the machines are provisioned in the catalog. Possible values are:

- PVS: Machine provisioned by PVS (machine may be physical, blade, VM...)
- MCS: Machine provisioned by MCS (machine must be a VM).

-- PvsAddress (System.String)

IP address of the PVS server to be used in a catalog with a PVS ProvisioningType.
-- PvsDomain (System.String)
The domain of the PVS server to be used in a catalog with a PVS ProvisioningType.

-- RemotePCDesktopGroupPriorities (System.Int32[])
Remote PC desktop groups' association priorities.

-- RemotePCDesktopGroupUids (System.Int32[])
UIDs of the Remote PC desktop groups associated with this catalog.

-- Scopes (Citrix.Broker Admin.SDK.ScopesReference[])
The list of the delegated admin scopes to which the catalog belongs.

-- SessionSupport (Citrix.Broker Admin.SDK.SessionSupport)
Specifies the session support of the machines in the catalog. Valid values are:
SingleSession, MultiSession.

-- Uid (System.Int32)
Uid of the catalog.

-- UnassignedCount (System.Int32)
The number of unassigned machines (machines not assigned to users).

-- UsedCount (System.Int32)
The number of machines in the catalog that are in a desktop group.

-- UUID (System.Guid)
The global ID of the catalog.

Related topics

New-BrokerCatalog
Remove-BrokerCatalog
Rename-BrokerCatalog
Set-BrokerCatalog

Parameters

-Uid <Int32>
Get catalogs with the specified UID.

| Required? | true |
### -Name (String)

Gets catalogs with the specified name.

| Required? | false |
| Default Value | 
| Accept Pipeline Input? | false |

### -AllocationType (AllocationType)

Gets catalogs that are of the specified allocation type. Values can be:

- Static - Machines in a catalog of this type are permanently assigned to a user.
- Permanent - equivalent to 'Static'.
- Random - Machines in a catalog of this type are picked at random and temporarily assigned to a user.

| Required? | false |
| Default Value | 
| Accept Pipeline Input? | false |

### -AssignedCount (Int32)

Gets catalogs containing a specified number of assigned machines (machines that have been assigned to users).

This property is typically used with advanced filtering; see about_Broker_Filtering.

| Required? | false |
| Default Value | 
| Accept Pipeline Input? | false |

### -AvailableAssignedCount (Int32)

Gets catalogs containing a specified number of available machines (those not in any desktop group) that are also assigned to users.

This property is typically used with advanced filtering; see about_Broker_Filtering.

<p>| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |</p>
<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
</table>

**AvailableCount (Int32)**

- Gets catalogs containing a specified number of available machines (those not in any desktop group).
- This property is typically used with advanced filtering; see about_Broker_Filtering.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int32</td>
<td>false</td>
</tr>
</tbody>
</table>

**AvailableUnassignedCount (Int32)**

- Gets catalogs containing a specified number of available machines (those not in any desktop group) that are not assigned to users.
- This property is typically used with advanced filtering; see about_Broker_Filtering.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int32</td>
<td>false</td>
</tr>
</tbody>
</table>

**Description (String)**

- Gets catalogs with the specified description.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>false</td>
</tr>
</tbody>
</table>

**HypervisorConnectionUid (Int32)**

- Gets catalogs associated with the specified hypervisor connection.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int32</td>
<td>false</td>
</tr>
</tbody>
</table>

**IsRemotePC (Boolean)**
Gets catalogs with the specified IsRemotePC value.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-MachinesArePhysical-<Boolean>

Specifies whether machines in the catalog can be power-managed by the Citrix Broker Service. Where the Citrix Broker Service cannot control the power state of the machine specify $true, otherwise $false. Can only be specified together with a provisioning type of Pvs or Manual, or if used with the legacy CatalogKind parameter only with Pvs or PvsPvd catalog kinds.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Metadata-<String>

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-MinimumFunctionalLevel-<FunctionalLevel>

Gets catalogs with a specific MinimumFunctionalLevel

Valid values are L5, L7

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-PersistUserChanges-<PersistUserChanges>

Specifies the location where the user changes will be persisted. Can only be set for single session catalogs. Values can be:

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.793
- **OnLocal** - User changes are persisted locally.
- **Discard** - User changes are discarded.
- **OnPvd** - User changes are persisted on the personal vDisk.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **ProvisioningSchemeId**<Guid>

  Gets catalogs associated with the specified provisioning scheme.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **ProvisioningType**<ProvisioningType>

  Specifies the provisioning type for the catalog. Values can be:
  - **Manual** - No provisioning.
  - **PVS** - Machine provisioned by PVS (machine may be physical, blade, VM...).
  - **MCS** - Machine provisioned by MCS (machine must be VM).

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **PvSAddress**<String>

  Gets catalogs containing machines provided by the Provisioning Services server with the specified address.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **PvsDomain**<String>

  Gets catalogs containing machines provided by the Provisioning Services server in the specified domain.
### -RemotePCDesktopGroupPriority<Int32>

Gets Remote PC catalogs with a Remote PC desktop group association with the specified priority.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -RemotePCDesktopGroupUid<Int32>

Gets Remote PC catalogs associated with the specified Remote PC desktop group.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -ScopeId<Guid>

Gets catalogs that are associated with the given scope identifier.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -ScopeName<String>

Gets catalogs that are associated with the given scope name.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -SessionSupport<SessionSupport>

Gets catalogs that have the specified session capability. Values can be:
- **SingleSession** - Single-session only machine.
- **MultiSession** - Multi-session capable machine.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>UnassignedCount</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>UsedCount</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>UUID</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>MachineUid</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **UnassignedCount** `<Int32>`

Gets catalogs containing a specified number of unassigned machines (machines not assigned to users).

This property is typically used with advanced filtering; see about_Broker_Filtering.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **UsedCount** `<Int32>`

Gets catalogs containing a specified number of machines used in a desktop group.

This property is typically used with advanced filtering; see about_Broker_Filtering.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **UUID** `<Guid>`

Get catalogs with the specified global ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **MachineUid** `<Int32>`

Gets the catalog containing the machine referenced by the specified unique identifier (UID).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
### -ReturnTotalRecordCount

A SwitchParameter.

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

| Required? | False |
| Default Value | False |
| Accept Pipeline Input? | False |

### -MaxRecordCount

A parameter of type: [Int32].

Specifies the maximum number of records to return.

| Required? | False |
| Default Value | 250 |
| Accept Pipeline Input? | False |

### -Skip

A parameter of type: [Int32].

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | False |
| Default Value | 0 |
| Accept Pipeline Input? | False |

### -SortBy

A parameter of type: [String].

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | False |
| Default Value | The default sort order is by name or unique identifier. |
| Accept Pipeline Input? | False |
-Filter <String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Property <String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None

Return Values

Citrix.BrokerAdmin.SDK.Catalog

Get-BrokerCatalog returns an object for each matching catalog.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerCatalog -AllocationType Random
C:\PS> Get-BrokerCatalog -Filter { AllocationType -eq 'Random'}

These commands return all catalogs containing machines that are randomly assigned to users. The second form uses advanced filtering (see about_Broker_Filtering).

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-BrokerCatalog -Filter { AvailableCount -gt 10}
This command returns catalogs with more than 10 unused machines that are available for assignment to users.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Get-BrokerCatalog -MaxRecordCount 1 -ProvisioningType Manual -SortBy '-AvailableCount'
This command returns the unmanaged catalog with the highest number of available machines.
Get-BrokerConfigurationSlot

Oct 10, 2013
Gets configuration slots configured for this site.

Syntax

Get-BrokerConfigurationSlot [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Get the list of configuration slots defined for this site. Each configuration slot determines a collection of related settings that can be specified in a machine configuration associated with this slot.

For example, a configuration slot may be defined to configure only "User Profile Manager" settings.

See about_Broker_Filtering for information about advanced filtering options.

-------------------------- BrokerConfigurationSlot Object

The configuration slot object returned represents a named collection of related settings.

-- Description (System.String)
Optional description of this configuration slot.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
A map of metadata associated with this configuration slot.

-- Name (System.String)
Unique name of this configuration slot.

-- SettingsGroup (System.String)
The encoded identity of the settings group that every setting in the associated machine configuration instances must belong to.

-- Uid (System.Int32)
Unique Uid of this configuration slot.

Related topics

New-BrokerConfigurationSlot
Remove-BrokerConfigurationSlot
Get-BrokerMachineConfiguration
Parameters

- **-Uid**<Int32>
  Get only the configuration slot with the specified unique identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Name**<String>
  Get only the configuration slot with the specified name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Metadata**<String>
  Gets records with matching metadata entries.

  The value being compared with is a concatenation of the key name, a colon, and the value. For example: `-Metadata "abc:x*"` matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-ReturnTotalRecordCount**<SwitchParameter>
  When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>False</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-MaxRecordCount**<Int32>
  Specifies the maximum number of records to return.
**-Skip**<br>-Int32-

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SortBy**<br>-String-

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter**<br>-String-

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Property**<br>-String[]-

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

None You cannot pipe input into this cmdlet.

**Return Values**

Citrix.BrokerAdmin.SDK.ConfigurationSlot

Get-BrokerConfigurationSlot returns an object for each matching slot.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

Get-ConfigurationSlot
Retrieves every configuration slot.

-------------------------- EXAMPLE 2 --------------------------

Get-ConfigurationSlot -Name "AppV"
Retrieves the configuration slot named "AppV".
Get-BrokerConfiguredFTA

Oct 10, 2013

Gets any file type associations configured for an application.

Syntax

Get-BrokerConfiguredFTA -Uid <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Gets any file type associations that are configured for content redirection to a published application.

File type association associates a file extension (such as ".txt") with an application (such as Notepad). In a Citrix environment file type associations on a user device can be configured so that when an user clicks on a document it launches the appropriate published application. This is known as "content redirection".

Configured file type associations are different from imported file type associations. Configured file type associations are those that are actually associated with published applications for the purposes of content redirection. Imported file type associations are lists of known file type associations for a given desktop group. See Update-BrokerImportedFTA for more information about imported file type associations.

-------------------------- BrokerConfiguredFTA Object

The BrokerConfiguredFTA object represents a file type association configured for a published application. It contains the following properties:

-- ApplicationUid (System.Int32)

The Uid of the application configured for the file type association.

-- ContentType (System.String)

Content type of the file, such as "text/plain" or "application/vnd.ms-excel".

-- ExtensionName (System.String)

A single file extension, such as .txt, unique within the scope of a desktop group.

-- HandlerDescription (System.String)

File type description, such as "Test Document", "Microsoft Word Text Document", etc.

-- HandlerName (System.String)

File type handler name, e.g. "Word.Document.8" or TXTFILE.

-- HandlerOpenArguments (System.String)
The arguments used for the ‘open’ action on files of this type.

-- Uid (System.Int32)

Unique internal identifier of configured file type association.

-- UUID (System.Guid)

UUID of the configured file type association.

Related topics

New-BrokerConfiguredFTA
Remove-BrokerConfiguredFTA
Update-BrokerImportedFTA

Parameters

-Uid<Int32>

Gets only the configured file type association for the specified unique identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ApplicationUid<Int32>

Gets only the configured file type associations for the specified application unique identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ContentType<String>

Gets only the configured file type associations for the specified content type (as seen in the Registry). For example, "text/plain" or "application/msword".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ExtensionName<String>
Gets only the configured file type associations for the specified extension name. For example, ".txt" or ".doc".

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **-HandlerDescription**<String>

Gets only the configured file type associations for the specified handler description. For example, "Text Document".

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **-HandlerName**<String>

Gets only the configured file type associations for the specified handler name. For example, "TXTFILE" or "Word.Document.8".

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **-HandlerOpenArguments**<String>

Gets only the configured file type associations for the specified open argument to the handler. For example, "%1".

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **-UUID**<Guid>

Gets configured file type associations with the specified value of UUID.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |
-ReturnTotalRecordCount (SwitchParameter)

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record contains additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

-MaxRecordCount (Int32)

Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |
| Accept Pipeline Input? | false |

-Skip (Int32)

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | false |

-SortBy (String)

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |
| Accept Pipeline Input? | false |

-Filter (String)

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.
Required? false
Default Value
Accept Pipeline Input? false

-Property<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Required? false
Default Value
Accept Pipeline Input? false

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

Required? false
Default Value Localhost. Once a value is provided by any cmdlet, this value will become the default.
Accept Pipeline Input? false

Input Type
None No input is accepted from the pipeline.

Return Values
Citrix.BrokerAdmin.SDK.ConfiguredFTA

This cmdlet returns one or more ConfiguredFTA objects.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerConfiguredFTA
Returns all configured file type associations.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-BrokerConfiguredFTA -ExtensionName ".txt"
Returns only configured file type associations that have a ".txt" extension.
Get-BrokerConnectionLog

Oct 10, 2013
Get entries from the site's session connection log.

Syntax

Get-BrokerConnectionLog [-Uid] <Int64> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Gets connection log entries matching the specified criteria. If no parameters are specified all connection log entries are returned.

The session connection log contains entries describing each brokered connection, or reconnection, attempt to a session in the site.

Each log entry describes a single connection brokering attempt to a new or existing session within the site. A single session can have multiple entries in the connection log, for example where the end user brokers a connection to a new session, disconnects and later brokers a reconnection. Conversely, other sessions may have none (e.g. console sessions).

By default connection log entries are removed after 48 hours.

For information about advanced filtering options when using the -Filter parameter, see about_Broker_Filtering; for information about machines, see about_Broker_Machines.

-------------------------- BrokerConnectionLog Object

The BrokerConnectionLog object represents a single brokered connection attempt to a new or existing session on a machine in the site. It contains the following properties:

-- BrokeringTime (System.DateTime)
The time at which the connection attempt was made.

-- BrokeringUserName (System.String)
The name of the user making the connection (in DOMAIN\User format).

-- BrokeringUserUPN (System.String)
The name of the user making the connection (in user@upndomain.com format).

-- ConnectionFailureReason (Citrix.Broker.Admin.SDK.ConnectionFailureReason?)
The status of the connection attempt. A value of None indicates that the connection was successfully established. $null

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.809
that the attempt is still in progress, and other values indicate that the attempt failed for the specified reason.

-- Disconnected (System.Boolean?)

Indicates if the connection was ended by disconnection (True), logoff or establishment failure (False), or is still active ($null).

-- EndTime (System.DateTime?)

The time at which the connection ended. If the connection ended by disconnection, the underlying machine session would still exist in a disconnected state.

-- EstablishmentTime (System.DateTime?)

The time at which the connection was successfully established. The value is $null if the connection attempt failed or is still in progress.

-- MachineDNSName (System.String)

The name of the machine to which the connection was made (in machine@dnsdomain.com form).

-- MachineName (System.String)

The name of the machine to which the connection was made (in DOMAIN\Machine format).

-- MachineUid (System.Int32)

The UID of the machine to which the connection was made.

-- Uid (System.Int64)

The UID of the connection log entry itself.

Related topics

Parameters

- Uid<Int64>

Gets a specific connection log entry identified by its UID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

- MachineName<String>

Gets connection log entries for the specified machines (in DOMAIN\Machine format).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
- **BrokeringTime**<br>Gets connection log entries with a specified brokering time. For more flexibility when searching on brokering time use the -Filter parameter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

- **BrokeringUserName**<br>Gets connection log entries for the specified users (in DOMAIN\User format).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

- **BrokeringUserUPN**<br>Gets connection log entries for the specified users (in user@upndomain.com format).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

- **ConnectionFailureReason**<br>Gets connection log entries which failed for the specified reason.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

- **Disconnected**<br>Gets connection log entries with the specified disconnection status, that is, whether the connection was disconnected, or logged-off.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>
-EndTime\<DateTime>\>

Gets connection log entries with the specified end time. For more flexibility when searching on end time use the -Filter parameter.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-EstablishmentTime\<DateTime>\>

Gets connection log entries with the specific establishment time. For more flexibility when searching on establishment time use the -Filter parameter.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-MachineDNSName\<String>\>

Gets connection log entries for the specified machines (in machine@dnsdomain.com format).

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-MachineUid\<Int32>\>

Gets connection log entries for a specific machine identified by its UID.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-ReturnTotalRecordCount\<SwitchParameter>\>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering
for details.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-MaxRecordCount&lt;Int32&gt;</td>
<td>false</td>
<td>250</td>
<td>false</td>
</tr>
<tr>
<td>-Skip&lt;Int32&gt;</td>
<td>false</td>
<td>0</td>
<td>false</td>
</tr>
<tr>
<td>-SortBy&lt;String&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-Filter&lt;String&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

Specifies the maximum number of records to return.

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.
-**Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.ConnectionLog

An entry from the connection log.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $when = [DateTime]::Now - [TimeSpan]::FromMinutes(30)
C:\PS> Get-BrokerConnectionLog -Filter {BrokeringTime -gt $when} -SortBy '+MachineName,-EndTime'

Gets all connection log entries for sessions brokered in the past 30 minutes, ordered first by machine name (ascending), then by session end time (descending).
Get-BrokerController

Oct 10, 2013
Gets Controllers running broker services in the site.

Syntax

Get-BrokerController [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Gets Controllers from the current site that match the specified search criteria.

Controllers are server machines running instances of the broker service. The broker service is responsible for the brokering of user sessions to desktops or applications, and for power management of the underlying machines. An operational site must contain at least one Controller.

If no search criteria are specified, all Controllers in the site are obtained.

-------------------------- BrokerController Object

The BrokerController object represents a single instance of a controller running instances of the broker service. It contains the following properties:

-- ActiveSiteServices (System.String[])
The services active on the site

-- AssociatedHypervisorConnectionUids (System.Int32[])
The UIDs of the associated hypervisor connections

-- ControllerVersion (System.String)
The version of the controller

-- DesktopsRegistered (System.Int32)
The count of the desktops registered on the controller

-- DNSName (System.String)
The DNS name of the controller

-- LastActivityTime (System.DateTime?)
The last reported as active time of the controller

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.815
-- LastStartTime (System.DateTime?)
The last start-up time of the controller

-- MachineName (System.String)
The windows name of the controller

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
The metadata for this command.

-- OSType (System.String)
The Operating System Type of the controller.

-- OSVersion (System.String)
The Operating System Version of the controller.

-- SID (System.String)
The SID of the controller.

-- State (Citrix.BrokerAdmin.SDK.ControllerState)
The state of the controller.

-- Uid (System.Int32)
The UID of the controller itself.

Related topics
Get-BrokerDesktop

Parameters

-Uid<Int32>

Gets only Controllers with the specified unique ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>True</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>False</td>
</tr>
</tbody>
</table>

-MachineName<String>

 Gets only Controllers with the specified Windows name. (domain\machine)

<table>
<thead>
<tr>
<th>Required?</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.816
Accept Pipeline Input?

- **ControllerVersion**<String>

  Gets only Controllers running the specified version of the broker service.

  - Required?
  - Default Value
  - Accept Pipeline Input?

- **DesktopsRegistered**<Int32>

  Gets only Controllers that have the specified number of desktops currently registered. This property is mainly of use with advanced filtering; see about_Broker_Filtering.

  - Required?
  - Default Value
  - Accept Pipeline Input?

- **DNSName**<String>

  Gets only Controllers with the specified DNS name ('machine.domain')

  - Required?
  - Default Value
  - Accept Pipeline Input?

- **LastActivityTime**<DateTime>

  Gets only Controllers last reported as active at the specified time. This property is mainly of use with advanced filtering; see about_Broker_Filtering.

  - Required?
  - Default Value
  - Accept Pipeline Input?

- **LastStartTime**<DateTime>

  Gets only Controllers that last started-up at the specified time. This property is mainly of use with advanced filtering; see about_Broker_Filtering.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Metadata**<String>

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **OSType**<String>

Gets only Controllers running the specified Operating System type.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **OSVersion**<String>

Gets only Controllers running the specified Operating System version.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **SID**<String>

Gets only Controllers with the specified SID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **State**<ControllerState>
Gets only Controllers currently in the specified state.

Valid values are: Failed, Off, On, and Active.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ReturnTotalRecordCount</td>
<td>false</td>
<td>False</td>
<td>false</td>
</tr>
<tr>
<td>-MaxRecordCount</td>
<td>false</td>
<td>250</td>
<td>false</td>
</tr>
<tr>
<td>-Skip</td>
<td>false</td>
<td>0</td>
<td>false</td>
</tr>
<tr>
<td>-SortBy</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-ReturnTotalRecordCount <SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

-MaxRecordCount<Int32>

Specifies the maximum number of records to return.

-Skip<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter <String>**

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Property <String[]>**

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress <String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

None You cannot pipe input into this cmdlet.

**Return Values**

Citrix.BrokerAdmin.SDK.Controller

Returns Controllers matching all specified selection criteria.

**Examples**

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerController -State Active
Gets all Controllers in the site that are currently active (powered on and fully operational).

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerController -Filter 'LastStartTime -gt "-30:00"'
Gets all Controllers in the site that started-up in the last 30 minutes.
Get-BrokerDBConnection

Oct 10, 2013
Gets the Citrix Broker Service's database connection string.

Syntax

```
Get-BrokerDBConnection [-AdminAddress <String>] [<CommonParameters>]
```

Detailed Description

Gets the database connection string from the currently selected Citrix Broker Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to a Broker SDK cmdlet.

Related topics

Set-BrokerDBConnection
Get-BrokerServiceStatus
Test-BrokerDBConnection

Parameters

- **-AdminAddress**<String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

Input Type

None You cannot pipe input into this cmdlet.

Return Values

System.String

The database connection string in use by the currently selected Citrix Broker Service instance.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerDBConnection -AdminAddress controller1.mydomain.net
Gets the database connection string in use by the Citrix Broker Service instance running on the controller with FQDN "controller1.mydomain.net".
Get-BrokerDBSchema

Oct 10, 2013

Gets SQL scripts to create or maintain the database schema for the Citrix Broker Service.

Syntax

Get-BrokerDBSchema -DatabaseName <String> [-ServiceGroupName <String>] [-ScriptType <DatabaseScriptType>] [-SID <String>] [-LocalDatabase] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Gets SQL scripts that can be used to create a new Citrix Broker Service database schema, add a new broker service to an existing site, remove a broker service from a site, or create a database server logon for a broker service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected broker service instance, otherwise the scripts relate to broker service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to a Broker SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines what script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to broker service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to broker service roles

If ScriptType is Evict, the returned script contains:

- Removal of broker service instance from database
o Removal of database user

If ScriptType is Login, the returned script contains:

o Creation of database server logon only

Related topics

Set-BrokerDBConnection

Test-BrokerDBConnection

Parameters

-**DatabaseName**<String>

Specifies the name of the database into which the new broker service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**ServiceGroupName**<String>

The name of the service group to be used when creating the Citrix Broker Service database schema. The service group is the collection of all broker services that share the same database and are considered equal (i.e. any service in the same service group can be used interchangeably).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**ScriptType**<DatabaseScriptType>

Specifies the type of database script to be returned. The different script types are as follows:

-- FullDatabase

Creates a database schema for the Citrix Broker Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

-- Instance

Adds a broker service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

-- Evict
Removes a broker service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

**-- Login**

Adds a logon for the broker service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;</td>
<td>false</td>
<td>None</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SID<String>**

Specifies the SID of the controller on which the broker service instance to remove from the database is running (only valid for a script type of Evict).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;</td>
<td>false</td>
<td>None</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LocalDatabase<SwitchParameter>**

Indicates that the database server is running on the same machine as the service instance itself. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Switch&gt;</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;</td>
<td>false</td>
<td>None</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.
Return Values

System.String

A string containing the required SQL script for applying to a database.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Get-BrokerDBSchema -DatabaseName MySiteDB -ServiceGroupName MyServiceGroup > C:\BrokerSchema.sql
Gets a script to create the full database schema for the Citrix Broker Service and copies it to a file called "C:\BrokerSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a broker service schema.

-------------------------- EXAMPLE 2 --------------------------

C:\PS>Get-BrokerDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\BrokerLogins.sql
Gets a script to create the appropriate database server logon for the broker service. This can be used when configuring a mirror server for use.
Get-BrokerDBVersionChangeScript

Oct 10, 2013
Gets an SQL service schema update script for the Citrix Broker Service.

Syntax

Get-BrokerDBVersionChangeScript -DatabaseName <String> -TargetVersion <Version> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Gets an SQL script that can be used to update the current Citrix Broker Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix Broker Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix Broker Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix Broker Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix Broker Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-BrokerServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Related topics

Get-BrokerInstalledDbVersion
Get-BrokerServiceStatus
Get-BrokerDBSchema

Parameters

-DatabaseName <String>
The name of the database containing the Citrix Broker Service schema to be updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-TargetVersion**<Version>

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

System.Management.Automation.PSCustomObject

The Get-BrokerDBVersionChangeScript cmdlet returns a PSCustomObject containing a script that can be used to update the Citrix Broker Service database schema. The object has the following properties:

-- **Script**

The raw text of the SQL script to apply the update.

-- **CanUndo**

If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained. Because Get-BrokerDBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.
-- NeedExclusiveAccess

If true, indicates that the update requires exclusive access to the Citrix Broker Service's schema while the update is applied; all Citrix Broker Services must be shutdown during the update.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $update = Get-BrokerDBVersionChangeScript -DatabaseName MySiteDB -TargetVersion 1.0.75.0
C:\PS> $update.Script > update_75.sql
Gets an SQL update script to update the current schema to version 1.0.75.0. The resulting update_75.sql script is suitable for direct use with the SQL Server SQLCMD utility.
Get-BrokerDelayedHostingPowerAction

Oct 10, 2013

Gets power actions that are executed after a delay.

Syntax

Get-BrokerDelayedHostingPowerAction [-Uid] <Int64> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Finds all delayed power actions that match the specified search criteria.

-------------------------- BrokerDelayedHostingPowerAction Object

The BrokerDelayedHostingPowerAction object represents an instance of a power action that is executed after a delay. It contains the following properties:


The power action to apply to the machine. Possible values are ShutDown and Suspend.

-- ActionDueTime (System.DateTime)

The UTC time at which the power action is due to be queued for execution.

-- DNSName (System.String)

The fully qualified DNS name of the machine that the power action applies to.

-- HostedMachineName (System.String)

The hypervisor's name for the machine that the power action applies to.

-- HypervisorConnectionUid (System.Int32)

The unique identifier of the hypervisor connection that is associated with the target machine.

-- MachineName (System.String)

The name of the machine that the power action applies to, in the form domain\machine.

-- Uid (System.Int64)

The unique identifier of the power action.

Related topics
New-BrokerDelayedHostingPowerAction

Remove-BrokerDelayedHostingPowerAction

Remove-BrokerHostingPowerAction

Parameters

- **Uid** <Int64>

  Gets only the single action record whose ID matches the specified value.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **MachineName** <String>

  Gets only the records for actions that are for machines whose name (of the form domain\machine) matches the specified string.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Action** <PowerManagementAction>

  Gets only the records for actions with the specified action type.

  Valid values are Shutdown and Suspend.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ActionDueTime** <DateTime>

  Gets only the records for actions due to be queued for execution at the specified time. This is useful with advanced filtering; for more information, see about_Broker_Filtering.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**-DNSName <String>**

Gets only the records for actions that are for machines whose DNS name matches the specified string.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-HostedMachineName <String>**

Gets only the records for actions that are for machines whose Hosting Name (the machine name as understood by the hypervisor) matches the specified string.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-HypervisorConnectionName <String>**

Gets only the records for actions for machines hosted through a hypervisor connection whose name matches the specified string.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-HypervisorConnectionUid <Int32>**

Gets only the records for actions for machines hosted through a hypervisor connection whose ID matches the specified value.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ReturnTotalRecordCount <SwitchParameter>**

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-MaxRecordCount<Int32>**

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Skip<Int32>**

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SortBy<String>**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter<String>**

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Property<String[]>**
Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type
None You cannot pipe input into this cmdlet.

Return Values
Citrix.BrokerAdmin.SDK.DelayedHostingPowerAction

Get-BrokerDelayedHostingPowerAction returns all delayed power actions that match the specified selection criteria.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerDelayedHostingPowerAction
Fetches records for all known delayed power actions that have not yet been queued for execution.
### Get-BrokerDesktop

**Oct 10, 2013**

Gets desktops configured for this site.

**Syntax**

```powershell
Get-BrokerDesktop [-Uid <Int32>] [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]
```

```

**Detailed Description**

This cmdlet is now deprecated, please use Get-BrokerMachine.

Retrieves desktops matching the specified criteria. If no parameters are specified this cmdlet enumerates all desktops.

Get-BrokerDesktop returns objects that combine desktop configuration and state information.

For single-session desktops, session information is displayed if present. It is possible that there are more than one sessions present on single-session desktops if ‘fast user switching’ is enabled, this cmdlet will prefer to return information about brokered sessions (rather than, for example, unbrokered direct RDP sessions). If there is no session running, session related fields return $null.

For multi-session desktops, no session information is ever displayed by this cmdlet, so session related fields always return $null. Get-BrokerSession can be used to get information about sessions on both multi-session and single-session desktops.

To count desktops, rather than retrieve full details of each desktop, use Group-BrokerDesktop instead.

For information about advanced filtering options, see about_Broker_Filtering; for information about desktops, see about_Broker_Desktops.

-------------------------- BrokerDesktop Object

The desktop object returned represents a physical or virtual machine configured in the site that is able to run either a Microsoft Windows desktop environment, individual applications, or both.

-- AgentVersion (System.String)

Version of the Citrix Virtual Delivery Agent (VDA) installed on the desktop.

-- ApplicationsInUse (System.String[])

List of applications in use on the desktop (in the form of browser name).

-- AssignedClientName (System.String)

The name of the endpoint client device that the desktop has been assigned to.

-- AssignedIPAddress (System.String)

The IP address of the endpoint client device that the desktop has been assigned to.

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.836
-- AssociatedUserFullNames (System.String[])

Full names of the users that have been associated with the desktop (in the form "Firstname Lastname").

Associated users are the current user(s) for shared desktops and the assigned users for private desktops.

-- AssociatedUserNames (System.String[])

Usernames of the users that have been associated with the desktop (usually in the form "domain\user").

Associated users are the current user(s) for shared desktops and the assigned users for private desktops.

-- AssociatedUserUPNs (System.String[])

The user principal names of the users that have been associated with the desktop (in the form user@upndomain.com).

Associated users are the current user(s) for shared desktops and the assigned users for private desktops.

-- AutonomouslyBrokered (System.Boolean?)

Session property indicating if the current session was started without the use of the broker.

Session properties are always null for multi-session desktops.

-- CatalogName (System.String)

Name of the catalog the machine is a member of.

-- CatalogUid (System.Int32)

UID of the catalog the machine is a member of.

-- ClientAddress (System.String)

Session property indicating the IP address of the client connected to the desktop.

Session properties are always null for multi-session desktops.

-- ClientName (System.String)

Session property indicating the host name of the client connected to the desktop.

Session properties are always null for multi-session desktops.

-- ClientVersion (System.String)

Session property indicating the version of the Citrix Virtual Delivery Agent running on the client connected.

Session properties are always null for multi-session desktops.

-- ColorDepth (Citrix.Broker.Admin.SDK.ColorDepth?)

The color depth setting configured on the desktop, possible values are:

$null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

-- ConnectedViaHostName (System.String)

Session property indicating the host name of the connection gateway, router or client.

Session properties are always null for multi-session desktops.

-- ConnectedViaIP (System.String)

Session property indicating the IP address of the connection gateway, router or client.

Session properties are always null for multi-session desktops.

-- ControllerDNSName (System.String)

The DNS host name of the controller that the desktop is registered to.
-- DeliveryType (Citrix.Broker.Admin.SDK.DeliveryType)
Denotes whether the desktop delivers desktops only, apps only or both.

-- Description (System.String)
Description of the desktop.

-- DesktopConditions (System.String[]) List of outstanding desktop conditions for the desktop.

-- DesktopGroupName (System.String)
Name of the desktop group the desktop has been assigned to.

-- DesktopGroupUid (System.Int32)
Uid of the desktop group the desktop has been assigned to.

-- DesktopKind (Citrix.BrokerAdminSDK.DesktopKind)
Deprecated. Denotes whether the desktop is private or shared. AllocationType should be used instead.

-- DeviceId (System.String)
Session property indicating a unique identifier for the client device that has most recently been associated with the current session.
Session properties are always null for multi-session desktops.

-- DNSName (System.String)
The DNS host name of the desktop.

-- FunctionalLevel (Citrix.BrokerAdminSDK.FunctionalLevel?)
The functional level of the desktop, if known.

-- HardwareId (System.String)
Session property indicating a unique identifier for the client hardware that has been most recently associated with the current session.
Session properties are always null for multi-session desktops.

-- HostedMachineId (System.String)
Unique ID within the hosting unit of the target managed desktop.

-- HostedMachineName (System.String)
The friendly name of a hosted desktop as used by its hypervisor. This is not necessarily the DNS name of the desktop.

-- HostingServerName (System.String)
DNS name of the hypervisor that is hosting the desktop if managed.

-- HypervisorConnectionName (System.String)
The name of the hypervisor connection that the desktop's hosting server is accessed through, if managed.

-- HypervisorConnectionUid (System.Int32?)
The UID of the hypervisor connection that the desktop's hosting server is accessed through, if managed.

-- IconUid (System.Int32?)
The UID of the desktop's icon that is displayed in StoreFront.

-- ImageOutOfDate (System.Boolean?)
Denotes whether the VM image for a hosted desktop is out of date.

-- InMaintenanceMode (System.Boolean)

Denotes whether the desktop is in maintenance mode.

-- IPAddress (System.String)

The IP address of the desktop.

-- IsAssigned (System.Boolean)

Denotes whether a private desktop has been assigned to a user/users, or a client name/address. Users can be assigned explicitly or by assigning on first use of the desktop.

-- IsPhysical (System.Boolean)

This value is true if the desktop is physical (i.e., not power managed by the Citrix Broker Service), and false otherwise.

-- LastConnectionFailure (Citrix.BrokerAdmin.SDK.ConnectionFailureReason)

The reason for the last failed connection between a client and the desktop.

-- LastConnectionTime (System.DateTime?)

Time of the last detected connection attempt that either failed or succeeded.

-- LastConnectionUser (System.String)

The SAM name (in the form DOMAIN\user) of the user that last attempted a connection with the desktop. If the SAM name is not available, the SID is used.

-- LastDeregistrationReason (Citrix.BrokerAdmin.SDK.DeregistrationReason?)

The reason for the last deregistration of the desktop with the broker. Possible values are:

AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

-- LastDeregistrationTime (System.DateTime?)

Time of the last deregistration of the desktop from the controller.

-- LastErrorReason (System.String)

The reason for the last error detected in the desktop.

-- LastErrorTime (System.DateTime?)

The time of the last detected error.

-- LastHostingUpdateTime (System.DateTime?)

Time of last update to any hosting data (such as power state) for this desktop reported by the hypervisor connection.

-- LaunchViaHostName (System.String)

Session property that denotes the host name of the StoreFront server used to launch the current brokered session.

Session properties are always null for multi-session desktops.

-- LaunchViaIP (System.String)

Session property that denotes the IP address of the StoreFront server used to launch the current brokered session.

Session properties are always null for multi-session desktops.

-- MachineInternalState (Citrix.BrokerAdmin.SDK.MachineInternalState)
The internal state of the machine associated with the desktop, reported while the desktop is registered to a controller, plus some private Citrix Broker Service states while the machine is not registered.

-- MachineName (System.String)
DNS host name of the machine associated with the desktop.

-- MachineUid (System.Int32)
Uid of the associated machine.

-- OSType (System.String)
A string that can be used to identify the operating system that is running on the desktop.

-- OSVersion (System.String)
A string that can be used to identify the version of the operating system running on the desktop, if known

-- PersistUserChanges (Citrix.BrokerAdmin.SDK.PersistUserChanges)
Describes whether/how the user changes are persisted. Possible values are:
- OnLocal - Persist the user changes on the local disk of the desktop.
- Discard - Discard user changes.
- OnPvd - Persist user changes on the Citrix Personal vDisk.

-- PowerActionPending (System.Boolean)
Property indicating whether there are any pending power actions for the desktop.

-- PowerState (Citrix.BrokerAdmin.SDK.PowerState)
The current power state of the desktop. Possible values are: Unmanaged, Unknown, Unavailable, Off, On, Suspending, TurningOn, TurningOff, Suspending, resuming.

-- Protocol (System.String)
Session property that denotes the protocol that the current session is using, can be either "HDX" or "RDP".

Session properties are always null for multi-session desktop.

-- ProvisioningType (Citrix.BrokerAdmin.SDK.ProvisioningType)
Describes how the machine associated with the desktop was provisioned, possible values are:
- PVS: Machine provisioned by PVS (may be physical, blade, VM,...)
- MCS: Machine provisioned by MCS (machine must be VM)

-- PublishedApplications (System.String[])
List of applications published by the desktop (displayed as browser names).

-- PublishedName (System.String)
The name of the desktop that is displayed in StoreFront, if the desktop is published.

-- PvdStage (Citrix.BrokerAdmin.SDK.PvdStage)
For a desktop supporting Personal vDisk technology (PvD), indicates the stage of the PvD image preparation.

-- RegistrationState (Citrix.BrokerAdmin.SDK.RegistrationState)
Indicates the registration state of the desktop. Possible values are: Unregistered, Initializing, Registered, AgentError.

-- SecureIcaActive (System.Boolean?)
Session property that indicates whether SecureICA is active on the current session.

Session properties are always null for multi-session desktops.

-- SecureIcaRequired (System.Boolean?)
Flag indicating whether SecureICA is required or not when starting a session on the desktop.

-- SessionHidden (System.Boolean?)
Session property that indicates if a session is hidden.

Session properties are always null for multi-session desktops.

-- SessionId (System.Int32?)
Deprecated. A unique identifier that Remote Desktop Services uses to track the session but it is only unique on that machine and only unique at any one particular time.

-- SessionState (Citrix.BrokerAdmin.SDK.SessionState?)
Session property indicating the state of the current session.

Session properties are always null for multi-session desktops, possible values are: Other, PreparingSession, Connected, Active,Disconnected, Reconnecting, NonBrokeredSession and Unknown.

-- SessionStateChangeTime (System.DateTime?)
Session property indicating the time of the last state change of the current session.

Session properties are always null for multi-session desktops.

-- SessionUid (System.Int64?)
Session property indicating the UID of the current session.

Session properties are always null for multi-session desktops.

-- SessionUserName (System.String)
Session property indicates the name of the current session's user (in the form DOMAIN\user).

Session properties are always null for multi-session desktops.

-- SessionUserSID (System.String)
Session property indicates the SID of the current session's user.

Session properties are always null for multi-session desktops.

-- SID (System.String)
The SID of the desktop.

-- SmartAccessTags (System.String[]) 
Session property that indicates the Smart Access tags for the current session.

Session properties are always null for multi-session desktops.

-- StartTime (System.DateTime?)
Session property that indicates the start time of the current session.

Session properties are always null for multi-session desktops.

-- SummaryState (Citrix.BrokerAdmin.SDK.DesktopSummaryState)
Indicates the overall state of the desktop. The overall state is a result of other more specific states such as session state, registration state and power state. Possible values: Off, Unregistered, Available, Disconnected, InUse, Preparing.

-- Tags (System.String[])
A list of tags for the desktop.

-- Uid (System.Int32)

UID of the desktop object.

-- WillShutdownAfterUse (System.Boolean)

Flag indicating whether this desktop is tainted and will be shut down after all sessions on the desktop have ended. This flag should only ever be true on power managed, single-session desktops.

Note: The desktop will not shut down if it is in maintenance mode, but will shut down after the desktop is taken out of maintenance mode.

Related topics

Group-BrokerMachine

Get-BrokerMachine

Parameters

-Uid<Int32>

Gets desktops with a specific UID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-MachineName<String>

Gets desktops with a specific machine name (in the form 'domain\machine').

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AgentVersion<String>

Gets desktops with a specific Citrix Virtual Delivery Agent version.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ApplicationInUse<String>

Gets desktops running a specified published application (identified by browser name).

String comparisons are case-insensitive.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AssignedClientName<String>

Gets desktops assigned to a specific client name.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>-AssignedIPAddress</strong> &lt;String&gt;</td>
<td>Gets desktops assigned to a specific client IP address.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-AssociatedUserFullName</strong> &lt;String&gt;</td>
<td>Gets desktops with an associated user identified by their full name (usually in the form 'first-name last-name').</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-AssociatedUserName</strong> &lt;String&gt;</td>
<td>Gets desktops with an associated user identified by their user name (in the form 'domain\user').</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-AssociatedUserUPN</strong> &lt;String&gt;</td>
<td>Gets desktops with an associated user identified by their User Principle Name (in the form 'user@domain').</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-AutonomouslyBrokered</strong> &lt;Boolean&gt;</td>
<td>Gets desktops with the autonomously brokered session flag.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-CatalogName</strong> &lt;String&gt;</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.843
Gets desktops from the catalog with the specific name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-CatalogUid<Int32>

Gets desktops from a catalog with a specific UID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ClientAddress<String>

Gets desktops with a specific client IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ClientName<String>

Gets desktops with a specific client name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ClientVersion<String>

Gets desktops with a specific client version.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ColorDepth<ColorDepth>

Gets desktops configured with a specific color depth.

Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ConnectedViaHostName<String>

Gets desktops with a specific host name of the incoming connection. This is usually a proxy or Citrix Access Gateway server.
-ConnectedViaIP<String>
Gets desktops with a specific IP address of the incoming connection.

-ControllerDNSName<String>
Gets desktops with a specific DNS name of the controller they are registered with.

-DeliveryType<DeliveryType>
Gets desktops of a particular delivery type.
Valid values are AppsOnly, DesktopsOnly, DesktopsAndApps

-Description<String>
Gets desktops with a specific description.

-DesktopCondition<String>
Gets desktop with an outstanding desktop condition condition.
Valid values are:
- CPU: Indicates the machine has high CPU usage
- ICALatency: Indicates the network latency is high
- UPMLogonTime: Indicates that the profile load time was high
- **DesktopGroupName** `<String>`
  
  Gets desktops from a desktop group with the specified name.
  
  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **DesktopGroupUid** `<Int32>`
  
  Gets desktops from a desktop group with the specified UID.
  
  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **DesktopKind** `<DesktopKind>`
  
  Deprecated: Use AllocationType parameter.

  Gets desktops of a particular kind.

  Valid values are Private, Shared.
  
  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **DeviceId** `<String>`
  
  Gets desktops with a specific client device ID.
  
  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **DNSName** `<String>`
  
  Gets desktops with a specific DNS name.
  
  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **FunctionalLevel** `<FunctionalLevel>`
  
  Gets desktops with a specific FunctionalLevel.

  Valid values are L5, L7.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |
**-HardwareId**<String>

Gets desktops with a specific client hardware ID.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-HostedMachineId**<String>

Gets desktops with a specific machine ID known to the hypervisor.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-HostedMachineName**<String>

Gets desktops with a specific machine name known to the hypervisor.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-HostingServerName**<String>

Gets desktops with a specific name of the hosting hypervisor server.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-HypervisorConnectionName**<String>

Gets desktops with a specific name of the hosting hypervisor connection.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-HypervisorConnectionUid**<Int32>

Gets desktops with a specific UID of the hosting hypervisor connection.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-IconUid**<Int32>

Gets desktops with a specific configured icon. Note that desktops with a null IconUid use the icon of the desktop group.
- **ImageOutOfDate**<Boolean>

  Gets desktops by whether their disk image is out of date (for machines provisioned using MCS only).

- **InMaintenanceMode**<Boolean>

  Gets desktops with a specific InMaintenanceMode setting.

- **IPAddress**<String>

  Gets desktops with a specific IP address.

- **IsAssigned**<Boolean>

  Gets desktops according to whether they are assigned or not. Desktops may be assigned to one or more users or groups, a client IP address or a client endpoint name.

- **IsPhysical**<Boolean>

  Specifies if machines in the catalog can be power managed by the Citrix Broker Service. Where the power state of the machine cannot be controlled, specify $true, otherwise $false. Can only be specified together with a provisioning type of Pvs or Manual, or if used with the deprecated CatalogKind parameter only with Pvs or PvsPvd catalog kinds.

- **LastConnectionFailure**<ConnectionFailureReason>

  Gets desktops with a specific reason for the last recorded connection failure. This value is None if the last connection was successful or if there has been no attempt to connect to the desktop yet.
Valid values are None, SessionPreparation, RegistrationTimeout, ConnectionTimeout, Licensing, Ticketing, and Other.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-LastConnectionTime</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-LastConnectionUser</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-LastDeregistrationReason</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-LastDeregistrationTime</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-LastErrorReason</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-LastConnectionTime:<DateTime>

Gets desktops that last connected at a specific time. This is the time that the broker detected that the connection attempt either succeeded or failed.

-LastConnectionUser:<String>

Gets desktops where a specific user name last attempted a connection (in the form 'domain\user').

-LastDeregistrationReason:<DeregistrationReason>

Gets desktops whose broker last recorded a specific deregistration reason. Valid values are $null, AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

-LastDeregistrationTime:<DateTime>

Gets desktops by the time that they were last deregistered.

-LastErrorReason:<String>

Gets desktops with the specified last error reason.
- **LastErrorTime**<br>**<DateTime>**<br>Gets desktops with the specified last error time.<br>---<br>Required? | false<br>---<br>Default Value<br>---<br>Accept Pipeline Input? | false<br>---<br>- **LastHostingUpdateTime**<br>**<DateTime>**<br>Gets desktops with a specific time that the hosting information was last updated.<br>---<br>Required? | false<br>---<br>Default Value<br>---<br>Accept Pipeline Input? | false<br>---<br>- **LaunchedViaHostName**<br>**<String>**<br>Gets desktops with a specific host name of the StoreFront server from which the user launched the session.<br>Session properties are always $null for multi-session desktops.<br>---<br>Required? | false<br>---<br>Default Value<br>---<br>Accept Pipeline Input? | false<br>---<br>- **LaunchedViaIP**<br>**<String>**<br>Gets desktops with a specific IP address of the StoreFront server from which the user launched the session.<br>Session properties are always $null for multi-session desktops.<br>---<br>Required? | false<br>---<br>Default Value<br>---<br>Accept Pipeline Input? | false<br>---<br>- **MachineInternalState**<br>**<MachineInternalState>**<br>Gets desktops with the specified internal machine state.<br>---<br>Required? | false<br>---<br>Default Value<br>---<br>Accept Pipeline Input? | false<br>---<br>- **MachineUid**<br>**<Int32>**<br>Gets desktops with a specific machine UID.<br>---<br>Required? | false<br>---<br>Default Value<br>---<br>Accept Pipeline Input? | false<br>---<br>- **OSType**<br>**<String>**
Gets desktops by the type of operating system they are running.

<table>
<thead>
<tr>
<th>Option</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-OSVersion</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

Gets desktops by the version of the operating system they are running.

<table>
<thead>
<tr>
<th>Option</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-OSVersion</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

Gets desktops by the location where the user changes are persisted.

- OnLocal - User changes are persisted locally.
- Discard - User changes are discarded.
- OnPvd - User changes are persisted on the Pvd.

<table>
<thead>
<tr>
<th>Option</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-PersistUserChanges</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

Gets desktops with a specific power action pending state.

Valid values are $true or $false.

<table>
<thead>
<tr>
<th>Option</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-PowerActionPending</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

Gets desktops with a specific power state.

Valid values are Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

<table>
<thead>
<tr>
<th>Option</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-PowerState</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

Gets desktops with connections using a specific protocol, for example 'HDX', or 'RDP'.

<table>
<thead>
<tr>
<th>Option</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Protocol</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
### ProvisioningType

**<ProvisioningType>**

Gets desktops that are in a catalog with a particular provisioning type. Values can be:

- **Manual** - No provisioning.
- **PVS** - Machine provisioned by PVS (machine may be physical, blade, VM, ...).
- **MCS** - Machine provisioned by MCS (machine must be VM).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
</tr>
</tbody>
</table>

### PublishedApplication

**<String>**

Gets desktops with a specific application published to them.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
</tr>
</tbody>
</table>

### PublishedName

**<String>**

Gets desktops with a specific published name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
</tr>
</tbody>
</table>

### PvdStage

**<PvdStage>**

Gets desktops with a specific personal vDisk stage. Valid values are None, Requested, Starting and Working.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
</tr>
</tbody>
</table>

### RegistrationState

**<RegistrationState>**

Gets desktops with a specific registration state. Valid values are Unregistered, Initializing, Registered and AgentError.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
</tr>
</tbody>
</table>

### SecureicaActive

**<Boolean>**
Gets desktops depending on whether the current session uses SecureICA or not.

Session properties are always $null for multi-session desktops.

<table>
<thead>
<tr>
<th>-SecureICARequired</th>
<th>Boolean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Gets desktops configured with a particular SecureICARequired setting. Note that the desktop setting of $null indicates that the desktop group value is used.

Session properties are always $null for multi-session desktops.

<table>
<thead>
<tr>
<th>-SessionHidden</th>
<th>Boolean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Gets desktops by whether their sessions are hidden or not. Hidden sessions are treated as though they do not exist when launching sessions; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

Session properties are always $null for multi-session desktops.

<table>
<thead>
<tr>
<th>-SessionId</th>
<th>Int32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Deprecated.

Gets desktops by session ID, a unique identifier that Remote Desktop Services uses to track the session but it is only unique on that machine.

<table>
<thead>
<tr>
<th>-SessionState</th>
<th>SessionState</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Gets desktops with a specific session state.

Valid values are $null, Other, PreparingSession, Connected, Active, Disconnected, Reconnecting, NonBrokeredSession, and Unknown.

Session properties are always $null for multi-session desktops.

<table>
<thead>
<tr>
<th>-SessionStateChangeTime</th>
<th>DateTime</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>-SessionUid</td>
<td>int64</td>
</tr>
<tr>
<td>-SessionUserName</td>
<td>String</td>
</tr>
<tr>
<td>-SessionUserSID</td>
<td>String</td>
</tr>
<tr>
<td>-SID</td>
<td>String</td>
</tr>
<tr>
<td>-SmartAccessTag</td>
<td>String</td>
</tr>
<tr>
<td>-StartTime</td>
<td>DateTime</td>
</tr>
</tbody>
</table>
Gets desktops with a specific session start time.
Session properties are always $null for multi-session desktops.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-**SummaryState**<DesktopSummaryState>

Gets desktops with a specific summary state.
Valid values are Off, Unregistered, Available, Disconnected, InUse and Preparing.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Tag**<String>

Gets desktops with a specific tag.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-**WillShutdownAfterUse**<Boolean>

Gets desktops depending on whether they shut down after use or not.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-**ApplicationUid**<Int32>

Gets desktops with a specific published application (identified by its UID).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-**ReturnTotalRecordCount**<SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-**MaxRecordCount**<Int32>
Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Skip<Int32>**

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SortBy<String>**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter<String>**

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Property<String[]>**

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.
Return Values

Citrix.BrokerAdmin.SDK.Desktop

Get-BrokerDesktop returns an object for each matching desktop.

Notes

To compare dates or times, use -Filter and relative comparisons. For more information, see about_Broker_Filtering and the examples.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerDesktop -RegistrationState Unregistered
C:\PS> Get-BrokerDesktop -Filter { RegistrationState -ne 'Registered' }
Both commands retrieve desktops that are unregistered. The second command also includes desktops with a registration state of AgentError.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-BrokerDesktop -SessionUid $null | ft -a DNSName,SummaryState
Gets desktops without sessions, listing the DNS name and current state.

-------------------------- EXAMPLE 3 --------------------------
C:\PS> Get-BrokerDesktop -Filter { OSType -like "Windows XP*" -and ImageOutOfDate }
Finds all Windows XP desktops with an out-of-date image.

-------------------------- EXAMPLE 4 --------------------------
C:\PS> Get-BrokerDesktop -ApplicationInUse "powerpoint"
Gets desktops running a published PowerPoint application. It matches any application browser name containing the word 'powerpoint'. String comparisons are case-insensitive.

-------------------------- EXAMPLE 5 --------------------------
C:\PS> Get-BrokerDesktop -DesktopCondition * DNSName,SessionUserName,DesktopConditions
Finds all desktops with an outstanding desktop condition, listing the affected desktop and user.

-------------------------- EXAMPLE 6 --------------------------
C:\PS> $d = (Get-Date).AddDays(-1)
C:\PS> Get-BrokerDesktop -Filter { StartTime -le $d } | ft MachineName,SessionUserName,StartTime,@{Label='Duration'; Expression={(Get-Date) - $_.StartTime}}
Finds users who have been logged on for more than a day, and outputs the machine name, start time, and duration the session has been logged on.
Get-BrokerDesktopGroup

Oct 10, 2013

Gets broker desktop groups configured for this site.

Syntax

Get-BrokerDesktopGroup [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Retrieve desktop groups matching the specified criteria. If no parameters are specified this cmdlet enumerates all desktop groups.

Desktop groups represent groups of desktops that are managed together for brokering purposes.

-------------------------- BrokerDesktopGroup Object

A desktop group object represents a collection of machines that are fully configured in a site that is able to run either a Microsoft Windows desktop environment, individual applications, or both.

--- AutomaticPowerOnForAssigned (System.Boolean)

Specifies whether assigned desktops in the desktop group are automatically started at the start of peak time periods. Only relevant for groups whose DesktopKind is Private.

--- AutomaticPowerOnForAssignedDuringPeak (System.Boolean)

Specifies whether assigned desktops in the desktop are automatically started throughout peak time periods. Only relevant for groups whose DesktopKind is Private and which have AutomaticPowerOnForAssigned set to true.

--- ColorDepth (Citrix.Broker.Admin.SDK.ColorDepth)
Default color depth of sessions started with machines in the desktop group. Possible values are:

FourBit, EightBit, SixteenBit, TwentyFourBit.

-- ConfigurationSlotUids (System.Int32[])

Uids of any configuration slots which hold machine configurations associated with the desktop group. The order of slot UIDs in this list correspond with the order of items in the associated MachineConfigurationNames and MachineConfigurationUids list properties, and so the same slot UID can appear more than once.

-- DeliveryType (Citrix.Broker.Admin.SDK.DeliveryType)

The type of resources being published. Possible values are:

DesktopsOnly, AppsOnly, DesktopsAndApps.

-- Description (System.String)

Description of the desktop group.

-- DesktopKind (Citrix.Broker.Admin.SDK.DesktopKind)

The kind of the desktops being published, possible values are:

Private and Shared.

-- DesktopsAvailable (System.Int32)

The number of machines in the desktop group in state Available; this is the number of machines with no sessions present.

-- DesktopsDisconnected (System.Int32)

The number of disconnected sessions present on machines in the desktop group.

-- DesktopsInUse (System.Int32)

The number of machines in the desktop group in state InUse; this is the number of machines with at least one session present.

-- DesktopsNeverRegistered (System.Int32)

The number of machines in the desktop group that have never registered with the current site.

-- DesktopsPreparing (System.Int32)

The number of machines in the desktop group whose PvD disk image is being prepared.

-- DesktopsUnregistered (System.Int32)

The number of machines in the desktop group that are currently unregistered.

-- Enabled (System.Boolean)

Specifies whether the desktop group is enabled or not; disabled desktop groups do not appear to users.

-- IconUid (System.Int32)
The Uid of the icon to be used as a default for desktops in the desktop group. Individual desktop objects can override this default by setting the IconUid parameter on the desktop object.

-- InMaintenanceMode (System.Boolean)
Specifies whether the machines in the desktop group are in maintenance mode or not.

-- IsRemotePC (System.Boolean)
Specifies whether the desktop group is a Remote PC desktop group.

-- MachineConfigurationNames (System.String[])
The MachineConfiguration names associated with the desktop group.

-- MachineConfigurationUids (System.Int32[])
The MachineConfiguration uids associated with the desktop group.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
Metadata associated with the desktop group.

-- MinimumFunctionalLevel (Citrix.BrokerAdmin.SDK.FunctionalLevel)
The minimum FunctionalLevel required for the machines in the desktop group to be able to register with the Citrix Broker Service.

-- Name (System.String)
Name of the desktop group.

-- OffPeakBufferSizePercent (System.Int32)
The percentage of machines that are kept available in an idle state outside peak hours.

-- OffPeakDisconnectAction (Citrix.BrokerAdmin.SDK.SessionChangeHostingAction)
The action that is performed after a configurable period of a user session disconnecting outside peak hours. Possible values are Nothing, Suspend or Shutdown.

-- OffPeakDisconnectTimeout (System.Int32)
The number of minutes before the configured action is performed after a user session disconnects outside peak hours.

-- OffPeakExtendedDisconnectAction (Citrix.BrokerAdmin.SDK.SessionChangeHostingAction)
The action performed after a second configurable period of a user session disconnecting outside peak hours. Possible values are Nothing, Suspend, or Shutdown.

-- OffPeakExtendedDisconnectTimeout (System.Int32)
The number of minutes before the second configured action is performed after a user session disconnects outside peak hours.
-- OffPeakLogOffAction (Citrix.BrokerAdmin.SDK.SessionChangeHostingAction)

The action performed after a configurable period of a user session ending outside peak hours. Possible values are Nothing, Suspend, or Shutdown.

-- OffPeakLogOffTimeout (System.Int32)

The number of minutes before the configured action is performed after a user session ends outside peak hours.

-- PeakBufferSizePercent (System.Int32)

The percentage of machines in the desktop group that are kept available in an idle state in peak hours.

-- PeakDisconnectAction (Citrix.BrokerAdmin.SDK.SessionChangeHostingAction)

The action performed after a configurable period of a user session disconnecting in peak hours. Possible values are Nothing, Suspend, or Shutdown.

-- PeakDisconnectTimeout (System.Int32)

The number of minutes before the configured action is performed after a user session disconnects in peak hours.

-- PeakExtendedDisconnectAction (Citrix.BrokerAdmin.SDK.SessionChangeHostingAction)

The action performed after a second configurable period of a user session disconnecting in peak hours. Possible values are Nothing, Suspend, or Shutdown.

-- PeakExtendedDisconnectTimeout (System.Int32)

The number of minutes before the second configured action is performed after a user session disconnects in peak hours.

-- PeakLogOffAction (Citrix.BrokerAdmin.SDK.SessionChangeHostingAction)

The action performed after a configurable period of a user session ending in peak hours. Possible values are Nothing, Suspend, or Shutdown.

-- PeakLogOffTimeout (System.Int32)

The number of minutes before the configured action is performed after a user session ends in peak hours.

-- ProtocolPriority (System.String[])

A list of protocol names in the order in which they are attempted for use during connection.

-- PublishedName (System.String)

The name of the desktop group as it is to appear to the user in StoreFront.

-- Scopes (Citrix.BrokerAdmin.SDK.ScopeReference[])

The list of the delegated admin scopes to which the desktop group belongs.

-- SecureIcaRequired (System.Boolean)

Flag that specifies if the SecureICA encryption of the HDX protocol is required for sessions of desktops in the desktop
group.

-- Sessions (System.Int32)

The total number of user sessions currently running on all of the machines in the desktop group.

-- SessionSupport (Citrix.BrokerAdmin.SDK.SessionSupport)

Specifies the session support (single/multi) of the machines in the desktop group. Machines with the incorrect session support for the desktop group will be unable to register with the Citrix Broker Service.

-- ShutdownDesktopsAfterUse (System.Boolean)

Specifies if the desktops will shut down after they have been used and there are no sessions running on the machine. The machines will not shut down if they are placed into maintenance mode, even if this flag is set to True. The machines, however, will shutdown after the machine is taken out of maintenance mode if the flag is still set.

-- Tags (System.String[])

Tags associated with the desktop group.

-- TimeZone (System.String)

The timezone that desktops in the desktop group are in (for power policy purposes).

-- TotalApplications (System.Int32)

Total number of applications associated with the desktop group.

-- TotalDesktops (System.Int32)

Total number of machines in the desktop group.

-- TurnOnAddedMachine (System.Boolean)

Specifies whether the broker should attempt to turn on power-managed machines when they are added to the desktop group.

-- Uid (System.Int32)

Uid of the desktop group.

-- UUID (System.Guid)

UUID if the desktop group.

Related topics

New-BrokerDesktopGroup
Set-BrokerDesktopGroup
Rename-BrokerDesktopGroup
Remove-BrokerDesktopGroup
Add-BrokerAdministrator

Add-BrokerUser

Add-BrokerTag

Parameters

- **Uid**<Int32>

Gets desktop groups with the specified value of Uid.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Name**<String>

Gets desktop groups whose name matches the supplied pattern.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AutomaticPowerOnForAssigned**<Boolean>

Gets only desktop groups with the specified value of AutomaticPowerOnForAssigned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AutomaticPowerOnForAssignedDuringPeak**<Boolean>

Gets only desktop groups with the specified value of AutomaticPowerOnForAssignedDuringPeak.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ColorDepth**<ColorDepth>

Gets only desktop groups with the specified color depth.
Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

<table>
<thead>
<tr>
<th>Field</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-DeliveryType</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Description</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-DesktopKind</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Enabled</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-**DeliveryType**<DeliveryType>

Gets desktop groups according to their delivery type.

Valid values are DesktopsOnly, AppsOnly and DesktopsAndApps.

<table>
<thead>
<tr>
<th>Field</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-DeliveryType</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Description</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-DesktopKind</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Enabled</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-**DesktopKind**<DesktopKind>

Gets desktops of a particular kind.

Valid values are Private and Shared.

<table>
<thead>
<tr>
<th>Field</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-DesktopKind</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Enabled</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-**Enabled**<Boolean>

Gets desktop groups with the specified value of Enabled.
Accept Pipeline Input?  |  false

**InMaintenanceMode** <Boolean>

Gets desktop groups with the specified value of InMaintenanceMode.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

**IsRemotePC** <Boolean>

Gets desktop groups with the specified IsRemotePC value.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

**Metadata** <String>

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

**MinimumFunctionalLevel** <FunctionalLevel>

Gets desktop groups with a specific MinimumFunctionalLevel.

Valid values are L5, L7

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

**OffPeakBufferSizePercent** <Int32>

Gets desktop groups with the specified value of OffPeakBufferSizePercent.
| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-OffPeakDisconnectAction**<SessionChangeHostingAction>

Gets desktop groups with the specified value of OffPeakDisconnectAction.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-OffPeakDisconnectTimeout**<Int32>

Gets desktop groups with the specified value of OffPeakDisconnectTimeout.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-OffPeakExtendedDisconnectAction**<SessionChangeHostingAction>

Gets desktop groups with the specified value of OffPeakExtendedDisconnectAction.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-OffPeakExtendedDisconnectTimeout**<Int32>

Gets desktop groups with the specified value of OffPeakExtendedDisconnectTimeout.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-OffPeakLogOffAction**<SessionChangeHostingAction>

Gets desktop groups with the specified value of OffPeakLogOffAction.
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-OffPeakLogOffTimeout</code> &lt;Int32&gt;</td>
<td>Gets desktop groups with the specified value of OffPeakLogOffTimeout.</td>
</tr>
<tr>
<td><code>-PeakBufferSizePercent</code> &lt;Int32&gt;</td>
<td>Gets desktop groups with the specified value of PeakBufferSizePercent.</td>
</tr>
<tr>
<td><code>-PeakDisconnectAction</code> &lt;SessionChangeHostingAction&gt;</td>
<td>Gets desktop groups with the specified value of PeakDisconnectAction.</td>
</tr>
<tr>
<td><code>-PeakDisconnectTimeout</code> &lt;Int32&gt;</td>
<td>Gets desktop groups with the specified value of PeakDisconnectTimeout.</td>
</tr>
<tr>
<td><code>-PeakExtendedDisconnectAction</code> &lt;SessionChangeHostingAction&gt;</td>
<td>Gets desktop groups with the specified value of PeakExtendedDisconnectAction.</td>
</tr>
</tbody>
</table>
-**PeakExtendedDisconnectTimeout**\(<\text{Int32}>\)

Gets desktop groups with the specified value of PeakExtendedDisconnectTimeout.

- **PeakLogOffAction**\(<\text{SessionChangeHostingAction}>\)

Gets desktop groups with the specified value of PeakLogOffAction.

- **PeakLogOffTimeout**\(<\text{Int32}>\)

Gets desktop groups with the specified value of PeakLogOffTimeout.

- **PublishedName**\(<\text{String}>\)

Gets desktop groups whose published name matches the supplied pattern.

- **ScopeId**\(<\text{Guid}>\)

Gets desktop groups that are associated with the given scope identifier.
| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-ScopeName**<String>

Gets desktop groups that are associated with the given scope name.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-SecureIcaRequired**<Boolean>

Gets desktop groups with the specified value of SecureIcaRequired.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-SessionSupport**<SessionSupport>

Gets desktop groups that have the specified session capability. Values can be:

- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-ShutdownDesktopsAfterUse**<Boolean>

Gets desktop groups with the specified value of ShutdownDesktopsAfterUse.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |
### -Tag<String>

Gets desktop groups tagged with the specified tag.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -TimeZone<String>

Gets desktop groups with the specified value of TimeZone.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -TotalApplications<Int32>

Gets desktop groups that are acting as delivery groups for the specified number of applications.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -TurnOnAddedMachine<Boolean>

Gets desktop groups with the specified value of TurnOnAddedMachine value.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -UUID<Guid>

Gets desktop groups with the specified value of UUID.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |
**-ApplicationUid<Int32>**

Gets desktop groups that publish the specified application (identified by Uid)

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-TagUid<Int32>**

Gets desktop groups to which the specified tag (identified by its Uid) has been added to help identify it - see Add-BrokerTag for more information.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-PowerTimeSchemeUid<Int32>**

Gets desktop groups associated with the specified power time scheme (identified by its Uid).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-MachineConfigurationUid<Int32>**

Gets desktop groups with the specified value of MachineConfiguration.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-RemotePCCatalogUid<Int32>**

Gets Remote PC desktop groups associated with the specified catalog.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
-ReturnTotalRecordCount [SwitchParameter]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-MaxRecordCount [Int32]

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Skip [Int32]

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-SortBy [String]

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Filter [String]

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.
-Property[String[]]

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Required? false
Default Value
Accept Pipeline Input? false

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

Required? false
Default Value Localhost. Once a value is provided by any cmdlet, this value will become the default.
Accept Pipeline Input? false

Input Type
None You cannot pipe input into this cmdlet.

Return Values
Citrix.BrokerAdmin.SDK.DesktopGroup

Get-BrokerDesktopGroup returns an object for each matching desktop group.

Notes
To perform greater-than or less-than comparisons, use -Filter. For more information, see about_Broker_Filtering and the examples.

Examples
-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerDesktopGroup -PublishedName EMEA*
Finds all desktop groups with published names starting with "EMEA".

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-BrokerDesktopGroup -InMaintenanceMode $true
Finds all desktop groups in maintenance mode.
Get-BrokerDesktopUsage

Oct 10, 2013
Get usage history of desktop groups.

Syntax


Detailed Description

Returns information, recorded by the broker on an hourly basis, about the number of desktops in use for each desktop group. Analyzing the historical usage records can give some guidance on usage patterns and help choosing idle pool settings.

Without parameters, Get-BrokerDesktopUsage returns the first 250 records. By using parameters, you can be more selective about the records that are returned.

To retrieve more than the default 250 records, use the -MaxRecordCount parameter. To select data for a specific desktop group, use either the -DesktopGroupName or -DesktopGroupUid parameters.

See the examples for this cmdlet and about_Broker_Filtering for details of how to perform advanced filtering.

-------------------------- BrokerDesktopUsage Object

Desktop usage object contains information to tell how many desktops in a desktop group are in use at a given time (identified by a timestamp).

-- DesktopGroupUid (System.Int32)
  Uid of the desktop group that the usage data corresponds to.

-- InUse (System.Int32)
  Specifies how many desktop are in use at the time the timestamp corresponds to.

-- Timestamp (System.DateTime)
  Date and time the desktop usage information corresponds to.

Related topics

Parameters

-DesktopGroupName <String>
  Gets usage records for the named desktop group or for multiple desktop groups if wildcards have been specified.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-DesktopGroupUid <Int32>
  Gets usage records for a specific desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-InUse <Int32>
  Gets usage records where the in-use count matches the specified value. This is useful when checking for zero or when used inside a -Filter expression.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-Timestamp <DateTime>
  Gets usage records where the timestamp matches the specified value.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
Gets usage records that occurred at the given time.

In general, Citrix recommends, using -Filter and relative comparisons. For a demonstration, see the examples.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ReturnTotalRecordCount</td>
<td>false</td>
<td>False</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-ReturnTotalRecordCount** `<SwitchParameter>`

When specified, this cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-MaxRecordCount <code>&lt;Int32&gt;</code></td>
<td>false</td>
<td>250</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-MaxRecordCount** `<Int32>`

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Skip <code>&lt;Int32&gt;</code></td>
<td>false</td>
<td>0</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Skip** `<Int32>`

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-SortBy <code>&lt;String&gt;</code></td>
<td>false</td>
<td>The default sort order is by name or unique identifier.</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-SortBy** `<String>`

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Filter <code>&lt;String&gt;</code></td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Filter** `<String>`

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Property <code>&lt;String[]&gt;</code></td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Property** `<String[]>`

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.
**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required?  | false |
| Default Value | localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**

None You cannot pipe objects to this cmdlet.

**Return Values**

Citrix.Broker.AdminSDK.DesktopUsage

Get-BrokerDesktopUsage returns an object for each matching record.

**Notes**

Desktop usage information is automatically deleted after 7 days.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

```powershell
C:\PS> Get-BrokerDesktopUsage -DesktopGroupName TestGroup -MaxRecordCount 24 -SortBy '-Timestamp' | ft -a @{Name='Time';Expression={'{0:t}' -f $_.Timestamp}},InUse
```

Returns the last 24 hours of usage information for desktop group TestGroup, formatting it as two columns labeled Time and InUse.

-------------------------- EXAMPLE 2 --------------------------

```powershell
C:\PS> $d = Get-Date -Hour 0 -Minute 0 -Second 0
C:\PS> Get-BrokerDesktopUsage -Filter { DesktopGroupName -eq TestGroup -and Timestamp -ge $d }
```

Returns today's usage information for desktop group TestGroup.

-------------------------- EXAMPLE 3 --------------------------

```powershell
C:\PS> $dg = Get-BrokerDesktopGroup TestGroup
C:\PS> Get-BrokerDesktopUsage -DesktopGroupUid $dg.Uid | Select Timestamp,InUse,@{Name='Percent';Expression={'{0:P0}' -f ($_.InUse / $dg.TotalDesktops)}}
```

Retrieves the usage history for desktop group TestGroup and adds a column showing the number of desktops in that group in use, as a percentage.
Get-BrokerEntitlementPolicyRule

Oct 10, 2013
Gets desktop rules from the site's entitlement policy.

Syntax

Get-BrokerEntitlementPolicyRule [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Returns desktop rules matching the specified search criteria from the site's entitlement policy. If no search criteria are specified, all desktop rules in the entitlement policy are obtained.

A desktop rule in the entitlement policy defines the users who are allowed per-session access to a machine from the rule's associated desktop group to run a full desktop session.

-------------------------- BrokerEntitlementPolicyRule Object

The BrokerEntitlementPolicyRule object represents a single desktop rule within the site's entitlement policy. It contains the following properties:

-- ColorDepth (Citrix.Broker.Admin.SDK.ColorDepth?)

The color depth of any desktop session launched by the user from the entitlement. If null, the equivalent setting from the rule's desktop group is used.

-- Description (System.String)

Optional description of the rule. The text may be visible to the end user, for example, as a tooltip associated with the desktop entitlement.

-- DesktopGroupUid (System.Int32)

The unique ID of the desktop group to which the rule applies.

-- Enabled (System.Boolean)

Indicates whether the rule is enabled. A disabled rule is ignored when evaluating the site's entitlement policy.

-- ExcludedUserFilterEnabled (System.Boolean)

Indicates whether the excluded users filter is enabled. If the filter is disabled then any user entries in the filter are ignored when entitlement policy rules are evaluated.

-- ExcludedUsers (Citrix.BrokerAdmin.SDK.ChbUser[])
The excluded users filter of the rule, that is, the users and groups who are explicitly denied an entitlement to a desktop session from this rule.

-- IconUid (System.Int32?)

The unique ID of the icon used to display the desktop entitlement to the user. If null, the equivalent setting from the rule's desktop group is used.

-- IncludedUserFilterEnabled (System.Boolean)

Indicates whether the included users filter is enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a desktop session by the rule.

-- IncludedUsers (Citrix.Broker.Admin.SDK.ChbUser[])

The included users filter of the rule, that is, the users and groups who are granted an entitlement to a desktop session by the rule.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

A collection of arbitrary key/value pairs that can be associated with the rule. The administrator can use these values for any purpose; they are not used by the site itself in any way.

-- Name (System.String)

The administrative name of the rule. Each rule in the site's entitlement policy must have a unique name (irrespective of whether they are desktop or application rules).

-- PublishedName (System.String)

The name of the desktop session entitlement as seen by the user. If null, the equivalent setting from the rule's desktop group is used.

-- SecureIcaRequired (System.Boolean?)

Indicates whether the rule requires the SecureICA protocol for desktop sessions launched using the entitlement. If null, the equivalent setting from the rule's desktop group is used.

-- Uid (System.Int32)

The unique ID of the rule itself.

-- UUID (System.Guid)

UUID of the rule.

Related topics

New-BrokerEntitlementPolicyRule
Set-BrokerEntitlementPolicyRule
Rename-BrokerEntitlementPolicyRule
Remove-BrokerEntitlementPolicyRule
Parameters

- **-Uid</Int32>**

Gets the desktop rule with the specified unique ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Name<String>**

Gets only desktop rules with the specified name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-ColorDepth<ColorDepth>**

Gets only desktop rules with the specified color depth.

Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Description<String>**

Gets only desktop rules with the specified description.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-DesktopGroupUid<Int32>**

Gets only desktop rules that apply to the desktop group with the specified unique ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-Enabled</strong>&lt;Boolean&gt;</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-ExcludedUser</strong>&lt;User&gt;</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-ExcludedUserFilterEnabled</strong>&lt;Boolean&gt;</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-IconUid</strong>&lt;Int32&gt;</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-IncludedUser</strong>&lt;User&gt;</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------</td>
</tr>
<tr>
<td><strong>-IncludedUserFilterEnabled</strong>&lt;Boolean&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>-Metadata</strong>&lt;String&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>-PublishedName</strong>&lt;String&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>-SecureIcaRequired</strong>&lt;Boolean&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>-UUID</strong>&lt;Guid&gt;</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><strong>-ReturnTotalRecordCount</strong></td>
<td>SwitchParameter</td>
</tr>
<tr>
<td><strong>-MaxRecordCount</strong></td>
<td>Int32</td>
</tr>
<tr>
<td><strong>-Skip</strong></td>
<td>Int32</td>
</tr>
<tr>
<td><strong>-SortBy</strong></td>
<td>String</td>
</tr>
</tbody>
</table>

**Required?** | false 

**Default Value** | False 

**Accept Pipeline Input?** | false 

- **ReturnTotalRecordCount**: This parameter, when specified, causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. It's useful for pagination of records or when you need to know how many records are available without loading all of them into memory. Setting this parameter also adds an error record to the output pipeline, which contains the count of records available. This is in addition to the objects returned by the cmdlet. For more details, refer to the about_Broker_Filtering documentation.

- **MaxRecordCount**: This parameter allows you to specify the maximum number of records you want to return from the cmdlet. It's a useful feature for limiting the amount of data that is processed and returned by the cmdlet, especially when dealing with large datasets.

- **Skip**: This parameter lets you skip a specified number of records before returning results. It's particularly useful when combined with the MaxRecordCount parameter to control the number of records written to the output pipeline. It also reduces the count returned by -ReturnTotalRecordCount, providing a more controlled flow of data.

- **SortBy**: This parameter allows you to sort the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. You can optionally prefix each name with a + or - to specify the order of sorting. Ascending order is assumed if no prefix is present. For example, sorting by a property named `Name` with ascending order would be `-SortBy Name`, whereas sorting it in descending order would be `-SortBy Name -`. This parameter helps organize the output according to your specific needs, making it easier to view and analyze the data.
Accept Pipeline Input? | false

**-Filter**<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**

None You cannot pipe input into this cmdlet.

**Return Values**

Citrix.BrokerAdmin.SDK.EntitlementPolicyRule

Get-BrokerEntitlementPolicyRule returns all desktop entitlement policy rules that match the specified selection criteria.

**Examples**

-------------- EXAMPLE 1 ---------------

C:\PS> Get-BrokerEntitlementPolicyRule
Returns all desktop rules from the entitlement policy. This offers a complete description of the current site's entitlement policy with respect to desktops published from shared desktop groups.
EXAMPLE 2

C:\PS> $dg = Get-BrokerDesktopGroup 'Customer Support'
C:\PS> Get-BrokerEntitlementPolicyRule -DesktopGroupUid $dg.Uid

Returns all desktop rules in the entitlement policy that give users entitlements to desktop sessions in the Customer Support desktop group.
Get-BrokerHostingPowerAction

Oct 10, 2013

Gets power actions queued for machines.

Syntax

Get-BrokerHostingPowerAction [-Uid] <Int64> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Finds power actions matching the specified search criteria from the queue of all known power actions. These power actions can be waiting to be dealt with or can be part way through being processed by the relevant hypervisor, or they can be recently completed actions. Completed actions are removed from the queue after a configured period, the default being one hour.

If no search criteria are specified all actions in the queue are obtained.

A Hosting Power Action record defines the action that is to be performed or has been performed, the machine that the action is to be applied to, the priority of the action in relation to other actions in the queue, times for points in the life of the action, and any results if the action has completed.

For a detailed description of the queuing mechanism, see 'help about_Broker_PowerManagement'.

-------------------------- BrokerHostingPowerAction Object

The BrokerHostingPowerAction object represents an instance of a power action. It contains the following properties:

-- Action (Citrix.BrokerAdmin.SDK.PowerManagementAction)

The power action to be performed. Possible values are: TurnOn, TurnOff, Shutdown, Reset, Restart, Suspend, Resume.

-- ActionCompletionTime (System.DateTime?)

The time when the power action was completed by the hypervisor connection.

-- ActionStartTime (System.DateTime?)

The time when the power action was started to be processed by the hypervisor.

-- ActualPriority (System.Int32)

The current priority of the operation after any priority boosting.

-- BasePriority (System.Int32)
The starting priority of the operation.

-- DNSName (System.String)

The fully qualified DNS name of the machine that the power action applies to.

-- FailureReason (System.String)

For failed power actions, an indication of the reason for the failure.

-- HostedMachineName (System.String)

The hypervisor's name for the machine that the power action applies to.

-- HypervisorConnectionUid (System.Int32)

The unique identifier of the hypervisor connection that is associated with the target machine.

-- MachineName (System.String)

The name of the machine that the power action applies to, in the form domain\machine.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

Metadata for this power action.

-- RequestTime (System.DateTime)

The timestamp of when the action was created and placed in the queue.

-- State (Citrix.BrokerAdmin.SDK.PowerActionState)

The current state of this power action. Possible values are: Pending, Started, Completed, Failed, Canceled, Deleted, Lost.

-- Uid (System.Int64)

The unique identifier of the power action.

Related topics

New-BrokerHostingPowerAction

Set-BrokerHostingPowerAction

Remove-BrokerHostingPowerAction

Parameters

-Uid<Int64>

Gets only the single action record whose ID matches the specified value.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? false

-MachineName<String>

Gets only the records for actions that are for machines whose name (of the form domain\machine) matches the specified string.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-Action<PowerManagementAction>

Gets only action records with the specified action type.

Valid values are TurnOn, TurnOff, ShutDown, Reset, Restart, Suspend and Resume.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-ActionCompletionTime<DateTime>

Gets only action records reported as having completed successfully at the specified time. This is useful with advanced filtering; for more information, see about_Broker_Filtering.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-ActionStartTime<DateTime>

Gets only action records reported as starting to be processed by the relevant hypervisor at the specified time. This is useful with advanced filtering; for more information, see about_Broker_Filtering.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-ActualPriority<Int32>

Gets only the records for actions whose current active priority matches the specified value.
<table>
<thead>
<tr>
<th>-<strong>Required?</strong></th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
</tbody>
</table>

**-BasePriority**<Int32>

Gets only the records for actions whose original priority matches the specified value.

<table>
<thead>
<tr>
<th>-<strong>Required?</strong></th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
</tbody>
</table>

**-DNSName**<String>

Gets only the records for actions that are for machines whose DNS name matches the specified string.

<table>
<thead>
<tr>
<th>-<strong>Required?</strong></th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
</tbody>
</table>

**-FailureReason**<String>

Gets only the records for actions that have failed and whose failure reason string matches the specified string.

<table>
<thead>
<tr>
<th>-<strong>Required?</strong></th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
</tbody>
</table>

**-HostedMachineName**<String>

Gets only the records for actions that are for machines whose Hosting Name (the machine name as understood by by the hypervisor) matches the specified string.

<table>
<thead>
<tr>
<th>-<strong>Required?</strong></th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
</tbody>
</table>

**-HypervisorConnectionName**<String>

Gets only the records for actions for machines hosted via a hypervisor connection whose name matches the specified string.
string.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-HypervisorConnectionUid<Int32>

Gets only the records for actions for machines hosted via a hypervisor connection whose ID matches the specified value.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Metadata<String>

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-RequestTime<DateTime>

Gets only the records for actions created and added to the queue at the specified time. This is useful with advanced filtering; for more information, see about_Broker_Filtering.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-State<PowerActionState>

Gets only the records for actions with the specified current state.

Valid values are Pending, Started, Completed, Failed, Canceled, Deleted and Lost.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**-ReturnTotalRecordCount** `<SwitchParameter>`

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-MaxRecordCount** `<Int32>`

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Skip** `<Int32>`

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SortBy** `<String>`

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
-Filter\<String>\>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Property<\String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress\<String>\>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.HostingPowerAction

Get-BrokerHostingPowerAction returns all power actions that match the specified selection criteria.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerHostingPowerAction

Fetches records for all known power actions either waiting to be processed, or currently being processed, or which have been processed in the last hour.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-BrokerHostingPowerAction -State Pending -HypervisorConnectionName 'XenPool5'
Fetches records for all power actions that are waiting to be processed and where the action is for a virtual machine that is hosted by the hypervisor called 'XenPool5'. 
Get-BrokerHypervisorAlert

Oct 10, 2013
Gets hypervisor alerts recorded by the controller.

Syntax

Get-BrokerHypervisorAlert -Uid <Int64> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

The Get-BrokerHypervisorAlert cmdlet gets alert objects reported by the hypervisors that the controller is monitoring.

Without parameters, Get-BrokerHypervisorAlert gets all of the alerts recorded. Use parameters to select which alerts are returned.

Once you have configured suitable alerts in your hypervisor, and configured the controller with your hypervisor details (see New-BrokerHypervisorConnection), the controller monitors each hypervisor for new alerts.

Four hypervisor alert metrics are recorded; these relate to the hypervisor host, not individual virtual machines:

-- Cpu: Reports excessive CPU usage.
-- Memory: Reports excessive memory usage.
-- Network: Reports high network activity.
-- Disk: Reports high disk activity.

Each alert also includes information about where and when the alert occurred, the severity of the alert (Red/Yellow), and the configuration of the triggered alert.

The following metrics are supported with these hypervisors:

-- VMware ESX (Cpu, Memory, Network, Disk)
-- Citrix XenServer (Cpu, Network)
-- Microsoft Hyper-V (None)

-------------------------- BrokerHypervisorAlert Object

The BrokerHypervisorAlert represents a hypervisor alert object. It contains the following properties:

-- HostingServerName (System.String)

The name of the hypervisor hosting this machine.
-- HypervisorConnectionUid (System.Int32)
The Uid of the hypervisor connection that reported this alert.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
Metadata for this hypervisor alert.

-- Metric (Citrix.BrokerAdmin.SDK.AlertMetric)
The metric this alert relates to: Cpu, Memory, Network or Disk.

-- Severity (Citrix.BrokerAdmin.SDK.AlertSeverity)
Severity of the alert (Red or Yellow). Red is more serious than Yellow.

-- Time (System.DateTime)
Time that the alert occurred.

-- TriggerInterval (System.TimeSpan?)
Number of ticks (100ns) before the alert can be raised again.

-- TriggerLevel (System.Double?)
Threshold level that the alert was configured to trigger at.

-- TriggerPeriod (System.TimeSpan?)
Duration the value was above the trigger level.

-- TriggerValue (System.Double?)
The value of the monitored metric that triggered the alert.

-- Uid (System.Int64)
The unique internal identifier of this alert.

Related topics

New-BrokerHypervisorConnection

Parameters

-Uid<Int64>

Gets the hypervisor alert with the specified UID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
-**HostingServerName**<String>

Gets alerts for the specified hosting hypervisor server.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-**HypervisorConnectionUid**<Int32>

Gets alerts for the specified hypervisor connection.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-**Metadata**<String>

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-**Metric**<AlertMetric>

Gets alerts for a specified metric.

Valid values are: Cpu, Memory, Network and Disk.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-**Severity**<AlertSeverity>

Gets alerts with the specified severity.

Valid values are: Red and Yellow.
- **Time**<DateTime>

  Gets alerts that occurred at a specific time.

  You can also use -Filter and relative comparisons; see the examples for more information.

- **TriggerInterval**<TimeSpan>

  Gets alerts with a specific trigger interval. This is the interval before the alert is raised again.

- **TriggerLevel**<Double>

  Gets alerts with a specific trigger threshold level.

- **TriggerPeriod**<TimeSpan>

  Gets alerts with a specific trigger period. This is the duration the threshold level was exceeded for, prior to the alert triggering.

- **TriggerValue**<Double>
Gets the value of the monitored metric that triggered the alert.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

(ReturnTotalRecordCount)<SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

(MaxRecordCount)<Int32>

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

(Skip)<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

(SortBy)<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
-Filter&lt;String&gt;

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-Property&lt;String[]&gt;

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AdminAddress&lt;String&gt;

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

Input Type

None You cannot pipe objects to this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.HypervisorAlert

Get-BrokerHypervisorAlert returns an object for each matching alert.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerHypervisorAlert -HostingServerName TestHyp* -Severity Red
Returns all serious (Red) alerts for any hosting server with a name that starts with 'TestHyp'.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerHypervisorAlert -Filter { Metric -eq 'Cpu' -and Time -ge '-1:0' }

Returns all CPU usage alerts that occurred in the last hour.
Get-BrokerHypervisorConnection

Oct 10, 2013
Gets hypervisor connections matching the specified criteria.

Syntax

Get-BrokerHypervisorConnection [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

The Get-BrokerHypervisorConnection cmdlet gets hypervisor connections matching the specified criteria. If no parameters are specified this cmdlet enumerates all hypervisor connections.

-------------------------- BrokerHypervisorConnection Object

The BrokerHypervisorConnection represents hypervisor connection object. It contains the following properties:

-- Capabilities (System.String[])
The set of capabilities as reported by the hypervisor.

-- HypHypervisorConnectionUid (System.Guid)
The Guid that identifies the hypervisor connection.

-- MachineCount (System.Int32)
Count of machines associated with this hypervisor connection.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
Collection of all the metadata associated to the hypervisor connection.

-- Name (System.String)
The display name of the hypervisor connection.

-- PreferredController (System.String)
The name of the controller which is preferred to be used, when available, to perform all communication to the hypervisor. The name is in DOMAIN\machine form. A preferred controller may have been automatically chosen when the hypervisor connection was created.

-- State (Citrix.BrokerAdmin.SDK.HypervisorConnectionState)
The state of the hypervisor connection.
-- Uid (System.Int32)
Unique internal identifier of hypervisor connection.

Related topics
New-BrokerHypervisorConnection
Remove-BrokerHypervisorConnection
Set-BrokerHypervisorConnection

Parameters
-Uid<Int32>
 Gets the hypervisor connection with the specified internal id.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-Name<String>
 Gets hypervisor connections with the specified name.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-HypHypervisorConnectionUid<Guid>
 Gets hypervisor connections with the specified Guid.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-MachineCount<Int32>
 Gets hypervisor connections with the specified machine count.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |
### Metadata

**Metadata**<String>

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: `-Metadata "abc:x*"` matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### PreferredController

**PreferredController**<String>

Gets hypervisor connections with the specified preferred controller. Specify the SAM name of the controller.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### State

**State**<HypervisorConnectionState>

Gets hypervisor connections with the specified connection state. Values can be:

- **Unavailable** - The broker is unable to contact the hypervisor.
- **InMaintenanceMode** - The hosting server is in maintenance mode.
- **On** - The broker is in contact with the hypervisor.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### ReturnTotalRecordCount

**ReturnTotalRecordCount**<SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | false

- **MaxRecordCount**<Int32>

Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |
| Accept Pipeline Input? | false |

- **Skip**<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | false |

- **SortBy**<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |
| Accept Pipeline Input? | false |

- **Filter**<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.
- **AdminAddress**: `<String>`

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

None

**Return Values**

Citrix.BrokerAdmin.SDK.HypervisorConnection

Get-BrokerHypervisorConnection returns an object for each matching hypervisor connection.

**Examples**

```
-------------------------- EXAMPLE 1 --------------------------
c:\PS> $hvConn = Get-BrokerHypervisorConnection -Name "hvConnectionName"
Gets a hypervisor connection by name.

-------------------------- EXAMPLE 2 --------------------------
c:\PS> $hvConn = Get-BrokerHypervisorConnection -PreferredController "domainName\controllerName"
Gets hypervisor connections by preferred controller.

-------------------------- EXAMPLE 3 --------------------------
c:\PS> $machine = Get-BrokerMachine -Uid $machineUid
c:\PS> $hvConn = Get-BrokerHypervisorConnection -Uid $machine.HypervisorConnectionUid
Gets hypervisor connection used by a (power managed) machine.
```
Get-BrokerIcon

Oct 10, 2013
Get stored icons.

Syntax

Get-BrokerIcon -Uid <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Reads a specific icon by Uid, or enumerates icons by passing no Uid.

--------------- BrokerIcon Object

The BrokerIcon object represents a single instance of an icon. It contains the following properties:

-- EncodedIconData (System.String)

The Base64 encoded .ICO format of the icon.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

Metadata for this command

-- Uid (System.Int32)

The UID of the icon itself.

Related topics

New-BrokerIcon
Remove-BrokerIcon

Parameters

-Uid <Int32>

Gets only the icon specified by unique identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Metadata <String>

Gets records with matching metadata entries.
The value being compared with is a concatenation of the key name, a colon, and the value. For example: `-Metadata "abc:x*"` matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

**-ReturnTotalRecordCount** `<SwitchParameter>`

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>False</td>
</tr>
</tbody>
</table>

**-MaxRecordCount** `<Int32>`

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>250</td>
</tr>
</tbody>
</table>

**-Skip** `<Int32>`

Skips the specified number of records before returning results. Also reduces the count returned by `-ReturnTotalRecordCount`.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>0</td>
</tr>
</tbody>
</table>

**-SortBy** `<String>`

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>default</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
**Default Value**
The default sort order is by name or unique identifier.

| Accept Pipeline Input? | false |

---

**Filter**<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

---

**Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

---

**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**
None

Input cannot be piped to this cmdlet.

**Return Values**
Citrix.BrokerAdmin.SDK.Icon

Retuns an Icon object for each enumerated icon.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerIcon
Enumerate all icons.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerIcon -Uid 1
Get the icon with Uid 1.
Get-BrokerImportedFTA

Oct 10, 2013
Gets the imported file type associations.

Syntax

Get-BrokerImportedFTA [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Returns the file type associations the system imports from worker machines.

File type association associates a file extension (such as ".txt") with an application (such as Notepad). In a Citrix environment file type associations on a user device can be configured so that when an user clicks on a document it launches the appropriate published application. This is known as "content redirection".

Imported file type associations are different from configured file type associations. Imported file type associations are lists of known file type associations for a given desktop group. Configured file type associations are those that are actually associated with published applications for the purposes of content redirection.

Initially the system is not aware of any extensions, and they must be imported by the Citrix administrator. See the Update-BrokerImportedFTA cmdlet for more information.

After file type extensions are imported, this cmdlet lets the administrator review which file type associations the system is aware of. ImportedFTA objects are also used when configuring content redirection. See the New-BrokerConfiguredFTA cmdlet for more information.

The imported file type associations are grouped according to the desktop group to which they belong, because the system expects all machines in the same desktop group to have the same file type associations. That may not be true, however, across desktop groups.

Note that the ImportedFTA object has several fields that are not currently used. Only those fields that are shared with the ConfiguredFTA object are actually used in some capacity.

-------------------------- BrokerImportedFTA Object

The BrokerImportedFTA object represents a file type association imported from worker machines. It contains the following properties:

--- ContentType (System.String)

Content type of the file, such as "text/plain" or "application/vnd.ms-excel".

--- Description (System.String)
File type description, such as "Test Document", "Microsoft Word Text Document", etc.

-- DesktopGroupUid (System.Int32)
The desktop group this file type belongs to.

-- Edit (System.String)
Edit command with full path to executable: "C:\Program Files (x86)\Microsoft Office\Office12\WINWORD.EXE" /n /dde

-- EditArguments (System.String)
The arguments used for the 'edit' action on files of this type. These are extracted from the full edit command, and may be empty.

-- EditExecutableName (System.String)
The executable name extracted from the Edit property, no path included. This is used for matching with published apps' executable when searching for the list of extensions an application is capable of handling.

-- ExtensionName (System.String)
A single file extension, such as .txt. Unique within the scope of a desktop group.

-- HandlerName (System.String)
File type handler name, e.g. "Word.Document.8" or TXTFILE.

-- Open (System.String)
Open command with full path to executable: "C:\Program Files (x86)\Microsoft Office\Office12\WINWORD.EXE" /n /dde

-- OpenArguments (System.String)
The arguments used for the 'open' action on files of this type. These are extracted from the full open command, and may be empty.

-- OpenExecutableName (System.String)
The executable name extracted from the Open property, no path included. This is used for matching with published apps' executable when searching for the list of extensions an application is capable of handling.

-- PerceivedType (System.String)
Perceived type, such as "text".

-- Print (System.String)
Print command: "C:\Program Files (x86)\Microsoft Office\Office12\WINWORD.EXE" /x /n /dde

-- PrintTo (System.String)
PrintTo command: "C:\Program Files (x86)\Microsoft Office\Office12\WINWORD.EXE" /n /dde

-- Uid (System.Int32)
Unique internal identifier of imported file type association.

Related topics
New-BrokerConfiguredFTA
Remove-BrokerImportedFTA
Update-BrokerImportedFTA

Parameters

- **-Uid**<Int32>
  Gets only the imported file type associations with the specified unique identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-ExtensionName**<String>
  Gets only the imported file type associations with the specified extension name. For example, ".txt" or ".png".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-ContentType**<String>
  Gets only the imported file type associations with the specified content type (as listed in the Registry). For example, "application/msword".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Description**<String>
  Gets only the imported file type associations with the specified description (as listed in the Registry). For example, "Text Document" or "Microsoft Word text document".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
## Accept Pipeline Input

- **Accept Pipeline Input?**
  - **Value:** false

## DesktopGroupId

- **-DesktopGroupUid<int32>**
  - Gets only the file type associations imported from a worker machine belonging to the specified desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

## Edit

- **-Edit<String>**
  - Gets only the imported file type associations with the specified Edit command, that includes both the executable name and path, and any arguments to that executable.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

## EditArguments

- **-EditArguments<String>**
  - Gets only the imported file type associations with the specified arguments to the Edit command.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

## EditExecutableName

- **-EditExecutableName<String>**
  - Gets only the imported file type associations with the specified executable for the Edit command.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

## HandlerName

- **-HandlerName<String>**
  - Gets only the imported file type associations with the specified handler name (as listed in the Registry). For example, "TXT FILE" or "Word.Document.8".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
- **Open**<String>

Gets only the imported file type associations with the specified Open command, that includes both the executable name and path, and any arguments to that executable.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **OpenArguments**<String>

Gets only the imported file type associations with the specified arguments to the Open command.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **OpenExecutableName**<String>

Gets only the imported file type associations with the specified executable for the Open command.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **PerceivedType**<String>

Gets only the imported file type associations with the specified perceived type (as listed in the Registry). For example, "document".

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **Print**<String>

Gets only the imported file type associations with the specified Print command.
### -PrintTo
- **Type:** `String`
- **Description:**
  
  Gets only the imported file type associations with the specified PrintTo command.

### -ReturnTotalRecordCount
- **Type:** `SwitchParameter`
- **Description:**
  
  When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker Filtering for details.

### -MaxRecordCount
- **Type:** `Int32`
- **Description:**
  
  Specifies the maximum number of records to return.

### -Skip
- **Type:** `Int32`
- **Description:**
  
  Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

### -SortBy
- **Type:** `String`

---

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>False</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>250</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter**<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

None This cmdlet does not accept any input from the pipeline.

**Return Values**

Citrix.BrokerAdmin.SDK.ImportedFTA
One or more ImportedFTA objects are returned as output.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerImportedFTA
Invoking this cmdlet with no arguments simply returns all of the imported file type association objects. By default, only the first 250 objects are returned. See the -MaxRecordCount and -Skip parameters for information about modifying this.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerImportedFTA -ExtensionName ".txt"
Retrieves all imported file type associations that have the extension ".txt". Note that because imported file type associations are per-desktop group, multiple instances may be returned.
Get-BrokerInstalledDbVersion

Oct 10, 2013
Gets Citrix Broker Service database schema version information.

Syntax

Get-BrokerInstalledDbVersion [-Upgrade] [-Downgrade] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Gets the current version number of the Citrix Broker Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-BrokerDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Related topics

Get-BrokerDBVersionChangeScript

Get-BrokerDBSchema

Parameters

-Upgrade <SwitchParameter>

Lists schema version numbers to which the current service schema could be upgraded.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-Downgrade <SwitchParameter>

Lists schema version numbers to which the current service schema could be downgraded.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |
-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type
None You cannot pipe input into this cmdlet.

Return Values
System.Version

Get-BrokerInstalledDBVersion returns database schema version numbers as requested.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerInstalledDBVersion
Gets the current Citrix Broker Service database schema version number.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerInstalledDBVersion -Upgrade
Gets the Citrix Broker Service database schema version numbers to which the service schema could be upgraded.
Get-BrokerMachine

Oct 10, 2013

Gets machines belonging to this site.

Syntax

Get-BrokerMachine [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Retrieves machines matching the specified criteria. If no parameters are specified, this cmdlet enumerates all machines.

Get-BrokerMachine returns objects that combine machine configuration and state information.

For single-session machines, session information is displayed if present. If "fast user switching" is enabled, more than one session may be present on single-session machines. Because this cmdlet returns information only for a single session, if two sessions are present it will return information about the brokered session (rather than, for example, an unbrokered direct RDP session). If there is no session running, session-related fields return $null.
For multi-session machines, no session information about single sessions is displayed by this cmdlet, and so are always $null. Get-BrokerSession can be used to get information about sessions on both multi-session and single-session machines.

To count machines, rather than retrieve full details of each machine, use Group-BrokerMachine instead.

See about_Broker_Filtering for information about advanced filtering options, and about_Broker_Machines for background information about machines.

-------------------------- BrokerMachine Object

The machine object returned represents a physical or virtual machine, which has been configured in the site.

-- AgentVersion (System.String)

Version of the Citrix Virtual Delivery Agent (VDA) installed on the machine.

-- AllocationType (Citrix.BrokerAdmin.SDK.AllocationType)

Describes how the machine is allocated to the user, can be Permanent or Random.

-- ApplicationsInUse (System.String[])

List of applications in use on the machine (in the form of browser name).

-- AssignedClientName (System.String)

The name of the endpoint client device that the machine has been assigned to.

-- AssignedIPAddress (System.String)

The IP address of the endpoint client device that the machine has been assigned to.

-- AssociatedUserFullNames (System.String[])

Full names of the users that have been associated with the machine (usually in the form "Firstname Lastname"). Associated users are the current user(s) for shared machines and the assigned users for private machines.

-- AssociatedUserNames (System.String[])

Usernames of the users that have been associated with the machine (in the form "domain\user"). Associated users are the current user(s) for shared machines and the assigned users for private machines.

-- AssociatedUserSIDs (System.String[])

The SIDs of the users that have been associated with the machine. Associated users are the current user(s) for shared machines and the assigned users for private machines.

-- AssociatedUserUPNs (System.String[])

The User Principal Names of the users associated with the machine (in the form user@domain). Associated users are the current user(s) for shared machines and the assigned users for private machines.
-- Capabilities (System.String[])  
List of the capabilities that the machine supports. Valid capabilities are:

- MultiSession: Indicates an RDS- (Terminal Services-) based machine, which supports multiple active sessions from different users.
- CBP1_5: Indicates the machine uses the CBP 1.5 protocol for communication.

-- CatalogName (System.String)  
Name of the catalog the machine is a member of.

-- CatalogUid (System.Int32)  
UID of the catalog the machine is a member of.

-- CatalogUUID (System.Guid)  
UUID of the catalog the machine is a member of.

-- ColorDepth (Citrix.Broker.Admin.SDK.ColorDepth?)  
The color depth setting configured on the machine, possible values are:

$null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

-- ControllerDNSName (System.String)  
The DNS host name of the controller that the machine is registered to.

-- DeliveryType (Citrix.Broker.Admin.SDK.DeliveryType?)  
Denotes whether the machine delivers desktops only, apps only or both.

-- Description (System.String)  
Description of the machine.

-- DesktopConditions (System.String[])  
List of outstanding desktop conditions for the machine.

-- DesktopGroupName (System.String)  
Name of the desktop group the machine is a member of.

-- DesktopGroupUid (System.Int32?)  
UID of the desktop group the machine is a member of.

-- DesktopGroupUUID (System.Guid?)  
UUID of the desktop group the machine is a member of.

-- DesktopKind (Citrix.BrokerAdmin.SDK.DesktopKind?)
Deprecated.

Denotes whether the machine is private or shared. Use AllocationType instead.

-- DesktopUid (System.Int32?)
The UID of the associated desktop object.

-- DNSName (System.String)
The DNS host name of the machine.

-- FaultState (Citrix.BrokerAdmin.SDK.MachineFaultState)
Summary state of any current fault state of the machine. Can be one of the following:
o None - No fault; machine is healthy.
o FailedToStart - Last power-on operation for machine failed.
o StuckOnBoot - Machine does not seem to have booted following power on.
o Unregistered - Machine has failed to register within expected period, or its registration has been rejected.
o MaxCapacity - Machine is reporting itself at maximum capacity.

-- FunctionalLevel (Citrix.BrokerAdmin.SDK.FunctionalLevel?)
Functional level of the machine, if known.

-- HostedMachineId (System.String)
Unique ID within the hosting unit of the target managed machine.

-- HostedMachineName (System.String)
The friendly name of a hosted machine as used by its hypervisor. This is not necessarily the DNS name of the machine.

-- HostingServerName (System.String)
DNS name of the hypervisor that is hosting the machine if managed.

-- HypervisorConnectionName (System.String)
The name of the hypervisor connection that the machine has been assigned to, if managed.

-- HypervisorConnectionUid (System.Int32?)
The UID of the hypervisor connection that the machine’s hosting server is accessed through.

-- HypHypervisorConnectionUid (System.Guid?)
The UUID of the hypervisor connection that the machine’s hosting server is accessed through

-- IconUid (System.Int32?)
The UID of the machine's icon that is displayed in StoreFront.

-- ImageOutOfDate (System.Boolean?)

Denotes if the VM image for a hosted machine is out of date.

-- InMaintenanceMode (System.Boolean)

Denotes if the machine is in maintenance mode.

-- IPAddress (System.String)

The IP address of the machine.

-- IsAssigned (System.Boolean)

Denotes whether a private desktop has been assigned to a user/users, or a client name/address. Users can be assigned explicitly or by assigning on first use of the machine.

-- IsPhysical (System.Boolean)

This value is true if the machine is physical (ie not power managed by the Citrix Broker service, and false otherwise.

-- LastConnectionFailure (Citrix.BrokerAdmin.SDK.ConnectionFailureReason)

The reason for the last failed connection between a client and the machine.

-- LastConnectionTime (System.DateTime?)

Time of the last detected connection attempt that either failed or succeeded.

-- LastConnectionUser (System.String)

The SAM name (in the form DOMAIN\user) of the user that last attempted a connection with the machine. If the SAM name is not available, the SID is used.

-- LastDeregistrationReason (Citrix.BrokerAdmin.SDK.DeregistrationReason?)

The reason for the last deregistration of the machine with the broker. Possible values are:

AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

-- LastDeregistrationTime (System.DateTime?)

Time of the last deregistration of the machine from the controller.

-- LastErrorReason (System.String)

The reason for the last error detected in the machine.
-- LastErrorTime (System.DateTime?)
The time of the last detected error.

-- LastHostingUpdateTime (System.DateTime?)
Time of last update to any hosting data (such as power states) for this machine reported by the hypervisor connection.

-- LoadIndex (System.Int32?)
Gives current effective load index for multi-session machines.

-- LoadIndexes (System.String[])
Gives the last reported individual load indexes that were used in the calculation of the LoadIndex value. Note that the LoadIndex value may have been subsequently adjusted due to session brokering operations. This value is only set for multi-session machines.

-- MachineInternalState (Citrix.BrokerAdmin.SDK.MachineInternalState)
The internal state of the machine; reported while the machine is registered to a controller, plus some private Citrix Broker Service states while the machine is not registered.

-- MachineName (System.String)
DNS host name of the machine.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
Any metadata that is associated with the machine.

-- OSType (System.String)
A string that can be used to identify the operating system that is running on the machine.

-- OSVersion (System.String)
A string that can be used to identify the version of the operating system running on the machine, if known.

-- PersistUserChanges (Citrix.BrokerAdmin.SDK.PersistUserChanges)
Describes if and how user changes are persisted. Possible values are:
- OnLocal - Persist the user changes on the local disk of the machine.
- Discard - Discard user changes.
- OnPvd - Persist user changes on the Citrix Personal vDisk.

-- PowerActionPending (System.Boolean)
Indicates if there are any pending power actions for the machine.

-- PowerState (Citrix.BrokerAdmin.SDK.PowerState)
The current power state of the machine. Possible values are: Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, resuming.

-- ProvisioningType (Citrix.Broker.Admin.SDK.ProvisioningType)
Describes how the machine was provisioned, possible values are:
o Manual: No automated provisioning.
o PVS: Machine provisioned by PVS (may be physical, blade, VM...)
o MCS: Machine provisioned by MCS (machine must be VM)

-- PublishedApplications (System.String[])
List of applications published by the machine (displayed as browser names).

-- PublishedName (System.String)
The name of the machine that is displayed in StoreFront, if the machine has been published.

-- PvdStage (Citrix.Broker.Admin.SDK.PvdStage)
For a machine supporting Personal vDisk technology (PvD), indicates the stage of the PvD image preparation.

-- RegistrationState (Citrix.Broker.Admin.SDK.RegistrationState)
Indicates the registration state of the machine. Possible values are: Unregistered, Initializing, Registered, AgentError.

-- ScheduledReboot (Citrix.Broker.Admin.SDK.ScheduledReboot)
Indicates the state of any scheduled reboot operation for a machine. Possible values:
o None: No reboot is scheduled.
o Pending: Machine is awaiting reboot but is available for use.
o Draining: Machine is awaiting reboot and is unavailable for new sessions; reconnections to existing connections are still allowed, however.
o InProgress: Machine is actively undergoing a scheduled reboot. o Natural: Natural reboot in progress. Machine is awaiting a restart.

-- SecureIcaRequired (System.Boolean?)
Flag indicating whether SecureICA is required or not when starting a session on the machine.

-- SessionAutonomouslyBrokered (System.Boolean?)
Session property indicating if the current session was started without the use of the broker.
Session properties are always null for multi-session machines.

-- SessionClientAddress (System.String)
Session property indicating the IP address of the client connected to the machine.
Session properties are always null for multi-session machines.

-- SessionClientName (System.String)
Session property indicating the host name of the client connected to the machine.

Session properties are always null for multi-session machines.

-- SessionClientVersion (System.String)
Session property indicating the host name of the client connected to the machine.

Session properties are always null for multi-session machine.

-- SessionConnectedViaHostName (System.String)
Session property indicating the host name of the connection gateway, router or client.

Session properties are always null for multi-session machines.

-- SessionConnectedViaIP (System.String)
Session property indicating the IP address of the connection gateway, router or client.

Session properties are always null for multi-session machines.

-- SessionCount (System.Int32)
Count of number of sessions on the machine.

-- SessionDeviceId (System.String)
Session property indicating a unique identifier for the client device that has most recently been associated with the current session.

Session properties are always null for multi-session machines.

-- SessionHardwareId (System.String)
Session property indicating a unique identifier for the client hardware that has been most recently associated with the current session.

Session properties are always null for multi-session machines.

-- SessionHidden (System.Boolean?)
Session parameter that indicates if a session is hidden.

Session parameters are always null for multi-session machines.

-- SessionKey (System.Guid?)
Session property indicating the key of the current session.

Session properties are always null for multi-session machines.
-- SessionLaunchedViaHostName (System.String)
Session property that denotes the host name of the StoreFront server used to launch the current brokered session. Session properties are always null for multi-session machines.

-- SessionLaunchedViaIP (System.String)
Session property that denotes the IP address of the StoreFront server used to launch the current brokered session. Session properties are always null for multi-session machines.

-- SessionProtocol (System.String)
Session property that denotes the protocol that the current session is using. Can be either "HDX" or "RDP". Session properties are always null for multi-session machines.

-- SessionSecureICAActive (System.Boolean?)
Session property that indicates whether SecureICA is active on the current session or not. Session properties are always null for multi-session machines.

-- SessionsEstablished (System.Int32)
Number of established sessions on this machine. For multi-session machines this excludes established sessions which have not yet completed their logon processing.

-- SessionSmartAccessTags (System.String[])
Session property that indicates the Smart Access tags for the current session. Session properties are always null on multi-session machines.

-- SessionsPending (System.Int32)
Number of pending (brokered but not yet established) sessions on this machine. For multi-session machines this also includes established sessions which have not yet completed their logon processing.

-- SessionStartTime (System.DateTime?)
Session property that indicates the start time of the current session. Session properties are always null on multi-session machines.

-- SessionState (Citrix.Broker.Admin.SDK.SessionState?)
Session property indicating the state of the current session, possible values are: Other, PreparingSession, Connected, Active, Disconnected, Reconnecting, NonBrokeredSession and Unknown. Session properties are always null for multi-session machines.

-- SessionStateChangeTime (System.DateTime?)
Session property indicating the time of the last state change of the current session.
Session properties are always null for multi-session machines.

-- SessionSupport (Citrix.Broker.Admin.SDK.SessionSupport)
Indicates the session support of the machine.
Possible values:
- SingleSession: Single-session only machine.
- MultiSession: Multi-session capable machine.

-- SessionUid (System.Int64?)
Session property indicating the UID of the current session.
Session properties are always null for multi-session machines.

-- SessionUserName (System.String)
Session property indicates the name of the current session's user (in the form DOMAIN\user).
Session properties are always null for multi-session machines.

-- SessionUserSID (System.String)
Session property indicates the SID of the current session's user.
Session properties are always null for multi-session machines.

-- SID (System.String)
The SID of the machine.

-- SummaryState (Citrix.Broker.Admin.SDK.DesktopSummaryState)
Indicates the overall state of the desktop associated with the machine. The overall state is a result of other more specific states such as session state, registration state and power state. Possible values: Off, Unregistered, Available, Disconnected, InUse, Preparing.

-- SupportedPowerActions (System.String[]) 
A list of power actions supported by this machine

-- Tags (System.String[])
A list of tags for the machine.

-- Uid (System.Int32)
UID of the machine object.

-- UUID (System.Guid)
UUID of the machine object.
-- VMToolsState (Citrix.BrokerAdmin.SDK.VMToolsState)
State of the hypervisor tools present on the VM (if any).
Possible values are:
NotPresent, Unknown, NotStarted, Running.

-- WillShutdownAfterUse (System.Boolean)
Flag indicating if this machine is tainted and will be shut down after all sessions on the machine have ended. This flag is only ever true on power-managed, single-session machines.
Note: The machine will not shut down if it is in maintenance mode; it will shut down only after it is taken out of maintenance mode.

-- WindowsConnectionSetting (Citrix.BrokerAdmin.SDK.WindowsConnectionSetting?)
The logon mode reported by Windows itself (multi-session machines only). For single-session machines the value is always hardwired to LogonEnabled.
Possible values are:
LogonEnabled, Draining, DrainingUntilRestart and LogonDisabled.

Related topics
Group-BrokerMachine

Parameters
- **-Uid**<Int32>
Gets a machine with a specific UID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-MachineName**<String>
Gets machines with a specific machine name (in the form domain\machine).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AgentVersion**<String>
Gets machines with a specific Citrix Virtual Delivery Agent version.

| **Required?** | **false** |
| **Default Value** | |
| **Accept Pipeline Input?** | **false** |

**-AllocationType**<AllocationType>

Gets machines from catalogs with the specified allocation type.

| **Required?** | **false** |
| **Default Value** | |
| **Accept Pipeline Input?** | **false** |

**-ApplicationInUse**<String>

Gets machines running a specified published application (identified by browser name).

String comparisons are case-insensitive.

| **Required?** | **false** |
| **Default Value** | |
| **Accept Pipeline Input?** | **false** |

**-AssignedClientName**<String>

Gets machines that have been assigned to the specific client name.

| **Required?** | **false** |
| **Default Value** | |
| **Accept Pipeline Input?** | **false** |

**-AssignedIPAddress**<String>

Gets machines that have been assigned to the specific IP address.

| **Required?** | **false** |
| **Default Value** | |
| **Accept Pipeline Input?** | **false** |
**-AssociatedUserFullName**<String>

Gets machines with an associated user identified by their full name (usually 'first-name last-name').

Associated users are all current users of a desktop, plus the assigned users for private desktops.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AssociatedUserName**<String>

Gets machines with an associated user identified by their user name (in the form 'domain\user').

Associated users are all current users of a desktop, plus the assigned users for private desktops.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AssociatedUserSID**<String>

Gets machines with an associated user identified by their Windows SID.

Associated users are all current users of a desktop, plus the assigned users for private desktops.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AssociatedUserUPN**<String>

Gets machines with an associated user identified by their User Principle Name (in the form 'user@domain').

Associated users are all current users of a desktop, plus the assigned users for private desktops.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-CatalogName**<String>

Gets machines from the catalog with the specific name.
-CatalogUid<Int32>

Gets machines from the catalog with the specific UID.

-CatalogUUID<Guid>

Gets machines from the catalog with the specific UUID.

-ColorDepth<ColorDepth>

Gets machines configured with a specific color depth.

Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

-ControllerDNSName<String>

Gets machines with a specific DNS name of the controller they are registered with.

-DeliveryType<DeliveryType>
Gets machines of a particular delivery type.

Valid values are AppsOnly, DesktopsOnly, DesktopsAndApps

- **Description**<String>
  Gets machines with a specific description.

- **DesktopCondition**<String>
  Gets machines with an outstanding desktop condition.
  Valid values are:
  - CPU: Indicates the machine has high CPU usage
  - ICALatency: Indicates the network latency is high
  - UPMLogonTime: Indicates that the profile load time was high

- **DesktopGroupName**<String>
  Gets machines from a desktop group with the specified name.

- **DesktopGroupUid**<Int32>
  Gets machines from a desktop group with a specific UID.
-DesktopGroupUUID<Guid>

Gets machines from a desktop group with a specific UUID.

-DesktopKind<DesktopKind>

Deprecated: Use AllocationType parameter.

Gets machines of a particular kind.

Valid values are Private, Shared.

-DesktopUid<Int32>

Gets the machine that corresponds to the desktop with the specific UID.

-DNSName<String>

Gets machines with the specific DNS name.
- **FaultState**<br>**<MachineFaultState>**<br>Gets machines currently in the specified fault state.<br><br>Required? | false<br>---|---<br>Default Value <br>Accept Pipeline Input? | false

- **FunctionalLevel**<br>**<FunctionalLevel>**<br>Gets machines with a specific FunctionalLevel.<br><br>Valid values are L5, L7<br><br>Required? | false<br>---|---<br>Default Value <br>Accept Pipeline Input? | false

- **HostedMachineId**<br>**<String>**<br>Gets machines with the specific machine ID known to the hypervisor.<br><br>Required? | false<br>---|---<br>Default Value <br>Accept Pipeline Input? | false

- **HostedMachineName**<br>**<String>**<br>Gets machines with the specific machine name known to the hypervisor.<br><br>Required? | false<br>---|---<br>Default Value <br>Accept Pipeline Input? | false

- **HostingServerName**<br>**<String>**<br>Gets machines by the name of the hosting hypervisor server.<br><br>Required? | false<br>---|---<br>Default Value
### Accept Pipeline Input?

- **Accept Pipeline Input?**
  - **false**

### HypervisorConnectionName

- **HypervisorConnectionName**<String>
  - **Required?** false
  - **Default Value**
  - **Accept Pipeline Input?** false

### HypervisorConnectionUid

- **HypervisorConnectionUid**<Int32>
  - **Required?** false
  - **Default Value**
  - **Accept Pipeline Input?** false

### HypHypervisorConnectionUid

- **HypHypervisorConnectionUid**<Guid>
  - **Required?** false
  - **Default Value**
  - **Accept Pipeline Input?** false

### IconUid

- **IconUid**<Int32>
  - **Required?** false
  - **Default Value**
  - **Accept Pipeline Input?** false

### ImageOutOfDate

- **ImageOutOfDate**<Boolean>
  - **Required?** false
  - **Default Value**

### HypervisorConnectionName

- **HypervisorConnectionName**
  - Gets machines with the specific name of the hypervisor connection hosting them.

- **Required?** false
- **Default Value**
- **Accept Pipeline Input?** false

### HypervisorConnectionUid

- **HypervisorConnectionUid**
  - Gets machines with the specific UID of the hypervisor connection hosting them.

- **Required?** false
- **Default Value**
- **Accept Pipeline Input?** false

### HypHypervisorConnectionUid

- **HypHypervisorConnectionUid**
  - Gets machines with the specific UUID of the hypervisor connection hosting them.

- **Required?** false
- **Default Value**
- **Accept Pipeline Input?** false

### IconUid

- **IconUid**
  - Gets machines by configured icon. Note that machines with a null IconUid use the icon of the desktop group.

- **Required?** false
- **Default Value**
- **Accept Pipeline Input?** false

### ImageOutOfDate

- **ImageOutOfDate**
  - Gets machines depending on whether their disk image is out of date or not (for machines provisioned using MCS only).

- **Required?** false
- **Default Value**
<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>false</th>
</tr>
</thead>
</table>

- **InMaintenanceMode**<Boolean>

Gets machines by whether they are in maintenance mode or not.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **IPAddress**<String>

Gets machines with a specific IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **IsAssigned**<Boolean>

Gets machines according to whether they are assigned or not. Machines may be assigned to one or more users or groups, a client IP address or a client endpoint name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **IsPhysical**<Boolean>

Gets machines according to whether they can be power managed by XenDesktop or not.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LastConnectionFailure**<ConnectionFailureReason>

Gets machines with a specific reason for the last recorded connection failure. This value is None if the last connection was successful or if there has been no attempt to connect to the desktop yet.

Valid values are None, SessionPreparation, RegistrationTimeout, ConnectionTimeout, Licensing, Ticketing, and Other.
- **LastConnectionTime** <DateTime>

  Gets machines on which a user session connection occurred at a specific time. This is the time at which the broker detected that the connection attempt either succeeded or failed.

- **LastConnectionUser** <String>

  Gets machines where a specific user name last attempted a connection (in the form 'domain\user').

- **LastDeregistrationReason** <DeregistrationReason>

  Gets machines whose broker last recorded a specific deregistration reason. Valid values are $null, AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultisessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

- **LastDeregistrationTime** <DateTime>

  Gets machines by the time that they were last deregistered.
<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

**-LastErrorReason**<String>

Gets machines with the specified last error reason.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

**-LastErrorTime**<DateTime>

Gets machines with the specified last error time.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

**-LastHostingUpdateTime**<DateTime>

Gets machines with a specific time that the hosting information was last updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoadIndex**<Int32>

Gets machines by their current load index.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

**-MachineInternalState**<MachineInternalState>

Gets machines with the specified internal state.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>Accept Pipeline Input?</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------</td>
</tr>
<tr>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-Metadata <String>

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-OSType <String>

Gets machines by the type of operating system they are running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-OSVersion <String>

Gets machines by the version of the operating system they are running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-PersistUserChanges <PersistUserChanges>

Gets machines by the location where the user changes are persisted.

- OnLocal - User changes are persisted locally.
- Discard - User changes are discarded.
- OnPvd - User changes are persisted on the Pvd.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------</td>
</tr>
</tbody>
</table>

**-PowerActionPending**<Boolean>

Gets machines depending on whether a power action is pending or not.

Valid values are True or False.

<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
</tbody>
</table>

**-PowerState**<PowerState>

Gets machines with a specific power state.

Valid values are Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
</tbody>
</table>

**-ProvisioningType**<ProvisioningType>

Gets machines that are in a catalog with a particular provisioning type. Values can be:

- Manual - No provisioning.
- PVS - Machine provisioned by PVS (machine may be physical, blade, VM,...).
- MCS - Machine provisioned by MCS (machine must be VM).

<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
</tbody>
</table>

**-PublishedApplication**<String>

Gets machines with a specific application published to them (identified by its browser name).

<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
</tbody>
</table>
- **PublishedName**<String>
  
  Gets desktops with a specific published name.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **PvdStage**<PvdStage>
  
  Gets machines at a specific personal vDisk stage.

  Valid values are None, Requested, PoweringOn and Working.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **RegistrationState**<RegistrationState>
  
  Gets machines in a specific registration state.

  Valid values are Unregistered, Initializing, Registered, and AgentError.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **ScheduledReboot**<ScheduledReboot>
  
  Gets machines according to their current status with respect to any scheduled reboots (for either scheduled desktop group reboots or image rollout purposes). Valid values are:

  - None - No reboot currently scheduled.
  - Pending - Reboot scheduled but machine still available for use.
  - Draining - Reboot scheduled. New logons are disabled, but reconnections to existing sessions are allowed.
  - InProgress - Machine is actively being rebooted. In Natural - Natural reboot in progress. Machine is awaiting a restart.

<p>| Required? | false |
| Default Value | |</p>
<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>false</th>
</tr>
</thead>
</table>

**-SecureIcaRequired**<Boolean>

Gets machines configured with a particular SecureIcaRequired setting. Note that the machine setting of $null indicates that the desktop group value is used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionAutonomouslyBrokered**<Boolean>

Gets machines with the autonomously brokered session flag.

Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionClientAddress**<String>

Gets machines with a specific client IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionClientName**<String>

Gets machines with a specific client name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionClientVersion**<String>

Gets machines with a specific client version.
- **SessionConnectedViaHostName**<String>
  Gets machines with a specific incoming connection host name. This is usually a proxy or Citrix Access Gateway server.
  Session properties are always null for multi-session machines.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **SessionConnectedViaIP**<String>
  Gets machines with a specific incoming connection IP address.
  Session properties are always null for multi-session machines.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **SessionCount**<Int32>
  Gets machines according to the total number of both pending and established user sessions on the machine.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **SessionDeviceId**<String>
  Gets machines with a specific client device ID.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **SessionHardwareId**<String>
Gets machines with a specific client hardware ID.

- **SessionHidden**<Boolean>

Gets machines depending on whether their sessions are hidden or not. Hidden sessions are treated as though they do not exist when launching sessions using XenDesktop; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

- **SessionKey**<Guid>

Gets machine running the session with a specified unique key.

Session properties are always null for multi-session machines.

- **SessionLaunchedViaHostName**<String>

Gets machines with a specific host name of the StoreFront server from which the user launched the session.

Session properties are always null for multi-session machines.

- **SessionLaunchedViaIP**<String>

Gets machines with a specific IP address of the StoreFront server from which the user launched the session.

Session properties are always null for multi-session machines.
### -SessionProtocol<br/>&lt;String&gt;

Gets machines with connections using a specific protocol, for example 'HDX', or 'RDP'.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -SessionSecureIcaActive<br/>&lt;Boolean&gt;

Gets machines depending on whether the current session uses SecureICA or not.

Session properties are always null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -SessionsEstablished<br/>&lt;Int32&gt;

Gets machines according to the number of established user sessions present on the machine.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -SessionSmartAccessTag<br/>&lt;String&gt;

Gets machines where the session has the specific SmartAccess tag.

Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**-SessionsPending**<Int32>

Get machines according to the number of pending user sessions for the machine.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionStartTime**<DateTime>

Gets machines with a specific session start time.

Session properties are always null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionState**<SessionState>

Get machines with a specific session state.

Valid values are $null, Other, PreparingSession, Connected, Active, Disconnected, Reconnecting, NonBrokeredSession, and Unknown.

Session properties are always null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionStateChangeTime**<DateTime>

Gets machines whose sessions last changed state at a specific time.

Session properties are always null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionSupport**<SessionSupport>
Gets machines that have the specified session capability. Values can be:

- **SingleSession** - Single-session only machine.
- **MultiSession** - Multi-session capable machine.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>-SessionUid</strong></td>
<td><code>&lt;Int64&gt;</code></td>
</tr>
<tr>
<td></td>
<td>Required?</td>
</tr>
<tr>
<td></td>
<td>Default Value</td>
</tr>
<tr>
<td></td>
<td>Accept Pipeline Input?</td>
</tr>
<tr>
<td><strong>-SessionUserName</strong></td>
<td><code>&lt;String&gt;</code></td>
</tr>
<tr>
<td></td>
<td>Required?</td>
</tr>
<tr>
<td></td>
<td>Default Value</td>
</tr>
<tr>
<td></td>
<td>Accept Pipeline Input?</td>
</tr>
<tr>
<td><strong>-SessionUserSID</strong></td>
<td><code>&lt;String&gt;</code></td>
</tr>
<tr>
<td></td>
<td>Required?</td>
</tr>
<tr>
<td></td>
<td>Default Value</td>
</tr>
<tr>
<td></td>
<td>Accept Pipeline Input?</td>
</tr>
<tr>
<td><strong>-SID</strong></td>
<td><code>&lt;String&gt;</code></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
Session properties are always null for multi-session machines.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>SummaryState</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

**SummaryState**<DesktopSummaryState>

Gets machines with a specific summary state.

Valid values are Off, Unregistered, Available, Disconnected, and InUse.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>SummaryState</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

**SupportedPowerActions**<String[]>

A list of power actions supported by this machine.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>SupportedPowerActions</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

**Tag**<String>

Gets machines where the session has the given SmartAccess tag.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

**UUID**<Guid>

Gets machines with the specified value of UUID.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>UUID</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
### -VMToolsState

**<VMToolsState>**

Gets machines with a specific VM tools state.

Valid values are NotPresent, Unknown, NotStarted, and Running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -WillShutdownAfterUse

**<Boolean>**

Gets machines depending on whether they shut down after use or not.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -WindowsConnectionSetting

**<WindowsConnectionSetting>**

Gets machines according to their current Windows connection setting (logon mode). Valid values are:

- LogonEnabled - All logons are enabled.
- Draining - New logons are disabled, but reconnections to existing sessions are allowed.
- DrainingUntilRestart - Same as Draining, but setting reverts to LogonEnabled when machine next restarts.
- LogonDisabled - All logons and reconnections are disabled.

This is a Windows setting and is not controlled by XenDesktop. It applies only to multi-session machines; for single-session machines its value is always LogonEnabled.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AssignedUserSID

**<String>**

Gets machines with the specific SID of the user to whom the desktop is assigned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

-ReturnTotalRecordCount <SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

| Required? | false |
| Default Value | False |

-MaxRecordCount <Int32>

Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |

-Skip <Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |

-SortBy <String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |

-Filter <String>
Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Property&lt;String[]&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Property&lt;String[]&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-AdminAddress&lt;String&gt;</td>
<td>false</td>
<td>localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.Machine

Get-BrokerMachine returns an object for each matching desktop.

Notes

It is generally better to compare dates and times using -Filter and relative comparisons. See about_Broker_Filtering and the examples in this topic for more information.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerMachine -PowerState Suspended
C:\PS> Get-BrokerMachine -Filter { PowerState -eq 'Suspended' }

These commands return all suspended machines. The second form uses advanced filtering (see about_Broker_Filtering).
-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerMachine -DNSName '*.mydomain.mycompany.com'
This command returns all machines belonging to the DNS domain mydomain.mycompany.com.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Get-BrokerMachine -Filter { RegistrationState -eq 'Registered' -and HypervisorConnectionUid -eq 5 }
This command returns all registered machines running on the specified hypervisor connection.

-------------------------- EXAMPLE 4 --------------------------

C:\PS> Get-BrokerMachine -MachineName 'MyDomain\X*' | Remove-BrokerDesktopGroup -DesktopGroup 2
This command finds all of the machines in MyDomain with names beginning with X and removes them from the specified desktop group.

-------------------------- EXAMPLE 5 --------------------------

C:\PS> Get-BrokerMachine -Filter { DesktopGroupUid -ne $null }
This command gets all desktops in a site. Use this instead of the deprecated Get-BrokerDesktop command.
Get-BrokerMachineCommand

Oct 10, 2013
Get the list of commands queued for delivery to a desktop.

Syntax

Get-BrokerMachineCommand -UId <Int64> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Get the list of commands queued for delivery to a desktop. Commands are batched and can be configured to be delivered at various times during a desktop session's lifetime. Normally commands are sent within a few minutes of being queued, but it is also possible to queue a command for a user who is not currently logged on or a desktop that is currently switched off.

See about_Broker_Filtering for information about advanced filtering options.

-------------------------- BrokerMachineCommand Object

The command object returned represents a command handled by a specific service on a desktop as determined by the Category property.

-- Category (System.String)

Category of the command.

-- CommandData (System.Byte[])

Additional binary data included when the command is sent.

-- CommandName (System.String)

Name of the command.

-- CompletionTime (System.DateTime?)

Time at which the command was sent, expired or canceled.

-- DesktopGroupNames (System.String[])

List of desktop group names that the command was restricted to.

-- MachineName (System.String)

Name of the machine this command is targeted at.

-- MachineUid (System.Int32?)

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
Unique ID of the machine this command is targeted at.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

Metadata for this command.

-- RequestTime (System.DateTime)

Time at which this command was created.

-- SendDeadline (System.TimeSpan)

Duration after which this command expires if it has not been sent yet.

-- SendDeadlineTime (System.DateTime?)

Time at which this command expires if it has not been sent yet.

-- SendTrigger (Citrix.BrokerAdmin.SDK.MachineCommandTrigger?)

Event that triggers the sending of the command. Valid values are NextContact, Broker, LogOn, Logoff, Disconnect and Reconnect.

-- SessionUid (System.Int64?)

Unique ID of the session this command is targeted at.

-- State (Citrix.BrokerAdmin.SDK.MachineCommandState)

Indicates whether the command is pending, sent, expired or canceled.

-- Synchronous (System.Boolean)

Flag that indicates if this is a synchronous command.

-- Uid (System.Int64)

Unique identifier of this machine command.

-- User (System.String)

Name of the user this command is targeted at.

Related topics

New-BrokerMachineCommand

Remove-BrokerMachineCommand

Parameters

-Uid<Int64>

Get only the command with the specified unique identifier.

| Required? | true |

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved. p.956
<table>
<thead>
<tr>
<th>-Category &lt;String&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Get only commands targeted to the specified service category.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-CommandName &lt;String&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Get only commands with the specified command name.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-CompletionTime &lt;DateTime&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Get only commands that entered the Sent, Failed, Canceled or Expired state at the specified time.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-MachineName &lt;String&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Get only commands targeted to the specified machine.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-MachineUid &lt;Int32&gt;</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Get only commands targeted to the specified machine.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
</tbody>
</table>
### Default Value

| Accept Pipeline Input? | false |

### Metadata

- **Metadata**: `<String>`

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: `-Metadata "abc:x*"` matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### RequestTime

- **RequestTime**: `<DateTime>`

Get only commands that were requested at the specified time.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### SendDeadline

- **SendDeadline**: `<TimeSpan>`

Get only commands that expire after the specified time span.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### SendDeadlineTime

- **SendDeadlineTime**: `<DateTime>`

Get only commands that have the specified deadline time.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### SendTrigger

- **SendTrigger**: `<MachineCommandTrigger>`

Get only commands that are due to be sent when the specified trigger occurs. Valid values are NextContact, Broker, LogOn.
Logoff, Disconnect and Reconnect.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-SessionUid</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-State</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-User</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-ReturnTotalRecordCount</td>
<td>false</td>
<td>False</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-SessionUid**<Int64>

Get only commands targeted to the specified session.

- **-State**<MachineCommandState>

Get only commands in the specified state. Valid values are Pending, Sent, Failed, Canceled and Expired.

- **-User**<String>

Get only commands targeted to the specified user.

- **-ReturnTotalRecordCount**<SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.
**-MaxRecordCount**<Int32>

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Skip**<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SortBy**<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter**<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

None No parameter is accepted from the input pipeline.

**Return Values**

`Citrix.BrokerAdmin.SDK.MachineCommand`

Returns Command objects matching all specified selection criteria.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

Get-BrokerMachineCommand
Returns all pending, canceled, expired and sent commands.

-------------------------- EXAMPLE 2 --------------------------

Get-BrokerMachineCommand -State Pending
Returns all queued commands.
Get-BrokerMachineConfiguration

Oct 10, 2013

Gets machine configurations defined for this site.

Syntax

Get-BrokerMachineConfiguration [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<<CommonParameters>>]


Detailed Description

Retrieves machine configurations matching the specified criteria. If no parameters are specified this cmdlet enumerates all machine configurations.

Machine configurations contain binary arrays of settings data that are managed using SDK snap-ins. Each machine configuration is associated with a configuration slot and referenced by Name. The configuration slot restricts the settings that can be held by the machine configuration. For example, only configurations for Citrix User Profile Manager can be associated with the "User Profile Manager" slot.

See about_Broker_Filtering for information about advanced filtering options.

-------------------------- BrokerMachineConfiguration Object

The machine configuration object returned represents a named collection of related settings values that are applied to a desktop group.

-- ConfigurationSlotUid (System.Int32)

Uid of the associated configuration slot.

-- Description (System.String)

Optional description of the machine configuration.

-- DesktopGroupUids (System.Int32[])

List of desktop group Uids that this machine configuration has been added to.

-- LeafName (System.String)

Name of this machine configuration.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

Map of metadata associated with this machine configuration.

-- Name (System.String)

Unique "SlotName\MachineConfigurationName" for this machine configuration.
**-- Policy (System.Byte[])**

A binary array of encoded settings.

**-- Uid (System.Int32)**

Uid of this machine configuration.

**Related topics**

- New-BrokerMachineConfiguration
- Set-BrokerMachineConfiguration
- Rename-BrokerMachineConfiguration
- Remove-BrokerMachineConfiguration
- Add-BrokerMachineConfiguration

**Parameters**

**-Uid<Int32>**

Get only the machine configuration with the specified unique identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Name<String>**

Get only the machine configuration with the specified name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LeafName<String>**

Get only the machine configurations that have the specified leaf name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Metadata<String>**


Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: `-Metadata "abc:x*"` matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **DesktopGroupUid** `<Int32>`

Get only the machine configurations that have been assigned to the specified desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ReturnTotalRecordCount** `<SwitchParameter>`

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **MaxRecordCount** `<Int32>`

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Skip** `<Int32>`

Skips the specified number of records before returning results. Also reduces the count returned by `-ReturnTotalRecordCount`.

| Required? | false |
-SortBy&lt;String&gt;

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |
| Accept Pipeline Input? | false |

-Filter&lt;String&gt;

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-Property&lt;String[]&gt;

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AdminAddress&lt;String&gt;

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |
None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.MachineConfiguration

Get-BrokerMachineConfiguration returns an object for each matching machine configuration.

Examples

-------------------------- EXAMPLE 1 --------------------------

Get-BrokerMachineConfiguration
Retrieves a list of every defined machine configuration.

-------------------------- EXAMPLE 2 --------------------------

Get-BrokerMachineConfiguration -Name Receiver\Engineering
Retrieves the machine configuration named "Receiver\Engineering".

-------------------------- EXAMPLE 3 --------------------------

Get-BrokerMachineConfiguration -Name UPM\*
Retrieves a list of every machine configuration associated with the configuration slot named "UPM".

-------------------------- EXAMPLE 4 --------------------------

Get-BrokerMachineConfiguration -LeafName "Dept"
Retrieves a list of every machine configuration with a LeafName that starts with "Dept", regardless of the associated configuration slot.
Get-BrokerMachineStartMenuShortcutIcon

Oct 10, 2013
Retrieves a Start Menu Shortcut icon from the specified machine.

Syntax
Get-BrokerMachineStartMenuShortcutIcon [-MachineName <String>] [-Path <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Retrieves the icon associated with a particular shortcut on a particular machine. This icon is usually used to help create a published application to access the shortcut.

Related topics
Get-BrokerMachine
New-BrokerIcon

Parameters
-MachineName <String>
Specify the name of the machine to use for icon retrieval for the specified shortcut path. The machine can be identified by DNS name, short name, SID, or name of the form domain\machine.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Path <String>
The location of the shortcut in the specified machine whose icon is being fetched.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type
None You cannot pipe input into this cmdlet.

Return Values
System.String

Get-BrokerMachineStartMenuShortcutIcon generates a Base64 encoded string containing the icon for the specified shortcut. This can be used as input to New-BrokerIcon cmdlet.

Examples

------------- EXAMPLE 1 -------------

C:\PS> $shortcuts = Get-BrokerMachineStartMenuShortcuts -MachineName 'MyDomain\MyMachine'
C:\PS> $encodedIconData = Get-BrokerMachineStartMenuShortcutIcon -MachineName 'MyDomain\MyMachine' -Path $shortcuts[0].ShortcutPath
C:\PS> $brokerIcon = New-BrokerIcon -EncodedIconData $encodedIconData
C:\PS> Set-BrokerApplication 'Notepad' -IconUid $brokerIcon.Uid

This example retrieves all Start Menu Shortcuts from 'MyDomain\MyMachine', and then the icon for the first shortcut from the returned list. The icon is then associated with a published application called 'Notepad'.
Get-BrokerMachineStartMenuShortcuts

Oct 10, 2013
Retrieves the Start Menu Shortcuts from the specified machine.

Syntax

Get-BrokerMachineStartMenuShortcuts [-MachineName] <String> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Retrieves the shortcuts defined for all the start menu items on a particular machine. The shortcuts obtained are from the 'All users' start menu; user-specific shortcuts are not found.

Related topics

Parameters

-MachineName <String>

Specify the name of the machine to use for shortcut retrieval. The machine can be identified by DNS name, short name, SID, or name of the form domain\machine.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

Input Type

None You cannot pipe input into this cmdlet.

Return Values

None or Citrix.BrokerAdmin.SDK.StartMenuShortcut


Examples
EXAMPLE 1

C:\PS> $shortcuts = Get-BrokerMachineStartMenuShortcuts -MachineName 'MyDomain\MyMachine'
This example retrieves all Start Menu Shortcuts from 'MyDomain\MyMachine'.

Get-BrokerPowerTimeScheme

Oct 10, 2013
Gets power management time schemes for desktop groups.

Syntax

Get-BrokerPowerTimeScheme [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Finds power time schemes matching the specified search criteria. Each desktop group in the site can have a number of power time schemes associated with it, and these time schemes control how the power states of machines in the group are managed.

If no search criteria are specified all power time schemes for all desktop groups are obtained.

Each power time scheme covers one or more days of the week, and defines which hours of those days are considered peak times and which are off-peak times. In addition, the time scheme defines a pool size value for each hour of the day for the days of the week covered by the time scheme. No one desktop group can be associated with two or more time schemes that cover the same day of the week.

For any day of the week not covered by any power time scheme, it is assumed that all hours are off-peak and no pool size management is required for any of the hours.

For more information about the power policy mechanism and pool size management, see 'help about_Broker_PowerManagement'.

-------------------------- BrokerPowerTimeScheme Object

The BrokerPowerTimeScheme object represents a power time scheme, defining peak/off-peak hours and idle pool sizes for desktop groups. It contains the following properties:

-- DaysOfWeek (Citrix.Broker.Admin.SDK.TimeSchemeDays)

The days of the week for which this scheme applies to (Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Weekdays, Weekend).

-- DesktopGroupUid (System.Int32)

The desktop group that this time scheme is for.

-- DisplayName (System.String)

The name of this time scheme, as displayed in the Studio console.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
Metadata for this power time scheme.

-- Name (System.String)

The unique name of this time scheme.

-- PeakHours (System.Boolean[])

A set of 24 boolean flag values, one for each hour of the day. The first value in the array relates to midnight to 00:59, the next one to 1 AM to 01:59 and so on, with the last array element relating to 11 PM to 11:59. If the flag is true it means that the associated hour of the day is considered a peak time; if it is false it means that it is considered off-peak.

-- PoolSize (System.Int32[])

A set of 24 integer values, one for each hour of the day. The first value in the array relates to midnight to 00:59, the next one to 1 AM to 01:59 and so on, with the last array element relating to 11 PM to 11:59. The value defines the number of machines (either as an absolute number or a percentage of the machines in the desktop group) that are to be maintained in a running state, whether they are in use or not. A value of -1 has special meaning: pool size management does not apply during such hours.

-- PoolUsingPercentage (System.Boolean?)

A boolean flag to indicate whether the integer values in the pool size array are to be treated as absolute values (if this value is false) or as percentages of the number of machines in the desktop group (if this value is true).

-- Uid (System.Int32)

Unique internal identifier of a time scheme.

Related topics

New-BrokerPowerTimeScheme
Set-BrokerPowerTimeScheme
Remove-BrokerPowerTimeScheme
Rename-BrokerPowerTimeScheme

Parameters

-Uid<int32>

Gets only the power time scheme with the specified Uid.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-Name<string>

Gets only power time schemes with the specified name.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-DesktopGroupUid</td>
<td>false</td>
<td>&lt;Int32&gt;</td>
<td>false</td>
</tr>
<tr>
<td>-Metadata</td>
<td>false</td>
<td>&lt;String&gt;</td>
<td>false</td>
</tr>
<tr>
<td>-ReturnTotalRecordCount</td>
<td>false</td>
<td>SwitchParameter</td>
<td>false</td>
</tr>
<tr>
<td>-MaxRecordCount</td>
<td>false</td>
<td>Int32</td>
<td>false</td>
</tr>
</tbody>
</table>

**-DesktopGroupUid (Int32)**

Gets only the power time schemes associated with the specified desktop group.

**-Metadata (String)**

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: `-Metadata "abc:x*"` matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

**-ReturnTotalRecordCount (SwitchParameter)**

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

**-MaxRecordCount (Int32)**

Specifies the maximum number of records to return.
-**Skip**<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**SortBy**<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Filter**<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.PowerTimeScheme

Get-BrokerPowerTimeScheme returns all power time schemes that match the specified selection criteria.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerPowerTime Scheme
Fetches all known power time schemes for all desktop groups in the site.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-BrokerPowerTimeScheme -DesktopGroupUid ( Get-BrokerDesktopGroup 'Sales Desktops').Uid
Fetches all the power time schemes for the desktop group called 'Sales Desktops'.
Get-BrokerPrivateDesktop

Oct 10, 2013
Get private desktops configured for this site.

Syntax

Get-BrokerPrivateDesktop [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

This cmdlet is deprecated, please use the Get-BrokerMachine cmdlet instead.

Retrieve private desktops matching the specified criteria. If no parameters are specified, this cmdlet enumerates all private desktops.

Get-BrokerPrivateDesktop returns configuration information only for private desktops (a DesktopKind of 'Private'). For more state information about desktops, or other types of desktop, use the Get-BrokerMachine cmdlet instead.

For information about advanced filtering options, see about_Broker_Filtering; for more information about desktops, see about_Broker_Desktops.

-------------------------- BrokerPrivateDesktop Object

Private desktops are machines that have been configured with a DesktopKind of 'Private'. They are allocated to either a user/users or a client name/address (but cannot be allocated to both).

--- AgentVersion (System.String)

Version of the Citrix Virtual Delivery Agent (VDA) installed on the desktop.

--- AssignedClientName (System.String)

Client name the desktop has been assigned to.

--- AssignedIPAddress (System.String)

IP Address the desktop has been assigned to.

--- ColorDepth (Citrix.BrokerAdmin.SDK.ColorDepth?)

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.976
The color depth setting configured on the desktop, possible values are:

$null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

-- ControllerDNSName (System.String)
The DNS host name of the controller that the desktop is registered to.

-- Description (System.String)
Description of the private desktop.

-- DesktopGroupUid (System.Int32)
Uid of the desktop group the desktop has been assigned to.

-- DNSName (System.String)
The DNS host name of the desktop.

-- HostedMachineId (System.String)
Unique ID within the hosting unit of the target managed desktop.

-- HostedMachineName (System.String)
The friendly name of a hosted desktop as used by its hypervisor. This is not necessarily the DNS name of the desktop.

-- HostingServerName (System.String)
DNS name of the hypervisor that is hosting the desktop if managed.

-- HypervisorConnectionUid (System.Int32?)
The UID of the hypervisor connection that the desktop has been assigned to, if managed.

-- IconUid (System.Int32?)
The UID of the desktop's icon that is displayed in StoreFront. If this is $null then the desktop will use the icon specified by the desktop group.

-- InMaintenanceMode (System.Boolean)
Denotes whether the desktop is in maintenance mode.

-- IPAddress (System.String)
The IP address of the desktop.

-- IsAssigned (System.Boolean)
Denotes whether a private desktop has been assigned to a user/users, or a client name/address. Users can be assigned explicitly or by assigning on first use of the desktop.

-- LastDeregistrationReason (Citrix.BrokerAdmin.SDK.DeregistrationReason?)
The reason for the last deregistration of the desktop with the broker. Possible values are:

AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed,
AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities,
MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion,
InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog,
FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate,
SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and
BrokerRegistrationLimitReached.

-- LastDeregistrationTime (System.DateTime?)
Time of the last deregistration of the desktop from the controller.

-- LastHostingUpdateTime (System.DateTime?)
Time of last update to any hosting data for this desktop reported by the hypervisor connection.

-- MachineName (System.String)
DNS host name of the machine associated with the desktop.

-- OSType (System.String)
A string that can be used to identify the operating system that is running on the desktop.

-- OSVersion (System.String)
A string that can be used to identify the version of the operating system running on the desktop, if known.

-- PowerState (Citrix.BrokerAdmin.SDK.PowerState)
The current power state of the desktop. Possible values are: Unmanaged, Unknown, Unavailable, Off, On, Suspended,
TurningOn, TurningOff, Suspending, Resuming.

-- PublishedName (System.String)
The name of the desktop that is displayed in StoreFront, if the desktop is published.

-- RegistrationState (Citrix.BrokerAdmin.SDK.RegistrationState)
Indicates the registration state of the desktop. Possible values are: Unregistered, Initializing, Registered, AgentError.

-- SecureIcaRequired (System.Boolean?)
Flag indicating whether SecureICA is required or not when starting a session on the desktop.

-- SID (System.String)
Security identifier of the private desktop.

-- Uid (System.Int32)
Unique identifier of the private desktop.
-- WillShutdownAfterUse (System.Boolean)

Flag indicating whether this desktop is tainted and will be shutdown after all sessions on the desktop have ended. This flag should only ever be true on power managed, single-session desktops.

Note: The desktop will not shut down if it is in maintenance mode, but will shut down after the desktop is taken out of maintenance mode.

Related topics

Set-BrokerPrivateDesktop

Parameters

- **Uid**<Int32>

Gets desktops by Uid.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **MachineName**<String>

Gets desktops by machine name (in the form 'domain\machine').

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AgentVersion**<String>

Gets desktops with a specific Citrix Virtual Delivery Agent (VDA) version.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AssignedClientName**<String>

Gets desktops assigned to a specific client name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------</td>
</tr>
</tbody>
</table>

**-AssignedIPAddress<String>**

Gets desktops assigned to a specific IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ColorDepth<ColorDepth>**

Gets desktops configured with a specific color depth.

Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ControllerDNSName<String>**

Gets desktops by the DNS name of the controller they are registered with.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Description<String>**

Gets desktops by description.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-DesktopGroupUid<int32>**

Gets desktops from a desktop group with a specific Uid.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-DNSName &lt;String&gt;</td>
<td>Gets desktops by DNS name.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>-HostedMachineId &lt;String&gt;</td>
<td>Gets desktops by the machine id known to the hypervisor.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>-HostedMachineName &lt;String&gt;</td>
<td>Gets desktops by the machine name known to the hypervisor.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>-HostingServerName &lt;String&gt;</td>
<td>Gets desktops by the name of the hosting hypervisor server.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>-HypervisorConnectionUid &lt;Int32&gt;</td>
<td>Gets desktops by the uid of the hosting hypervisor connection.</td>
</tr>
</tbody>
</table>
### -IconUid<Int32>

Gets desktops by configured icon. Note that desktops with a $null IconUid use the icon of the desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -InMaintenanceMode<Boolean>

Gets desktops by the InMaintenanceMode setting.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -IPAddress<String>

Get desktops by their IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -IsAssigned<Boolean>

Gets desktops depending on whether they are assigned or not. Private desktops can be assigned to either a user/users or client names/addresses.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -LastDeregistrationReason<DeregistrationReason>

Gets desktops whose broker last recorded a specific deregistration reason.

Valid values are $null, AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion,
AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LastDeregistrationTime**<DateTime>

Gets desktops by the time that they were last deregistered.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LastHostingUpdateTime**<DateTime>

Gets desktops by the time that the hosting information was last updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-OSType**<String>

Gets desktops by the type of operating system they are running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-OSVersion**<String>

Gets desktops by the version of the operating system they are running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**Default Value**

| Accept Pipeline Input? | false |

---

**-PowerState</PowerState>**

Gets desktops by power state.

Valid values are Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

---

**-PublishedName</String>**

Gets desktops by published name.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

---

**-RegistrationState</RegistrationState>**

Gets desktops by registration state.

Valid values are Registered, Unregistered, and AgentError.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

---

**-SecureIcaRequired</Boolean>**

Gets desktops configured with a particular SecureIcaRequired setting. Note that the desktop setting of $null indicates that the desktop group value is used.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

---

**-SID</String>**
Gets desktops by machine SID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Tag<String>

Gets desktops tagged with the given tag.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-WillShutdownAfterUse<Boolean>

Gets desktops depending on whether they will be automatically shut down when the current session ends or not.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AssignedUserSID<String>

Gets desktops with the given assigned user (specified by SID).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ReturnTotalRecordCount<br>SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
-MaxRecordCount<Int32>

Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |
| Accept Pipeline Input? | false |

-Skip<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | false |

-SortBy<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |
| Accept Pipeline Input? | false |

-Filter<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-Property<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

| Required? | false |
### -AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

#### Notes

To compare dates/times, use -Filter and relative comparisons. See about_Broker_Filtering for details.

#### Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $list = 'Unmanaged','On','TurningOn','Resuming'
C:\PS> Get-BrokerPrivateDesktop -Filter { PowerState -in $list } | ft -a DNSName,PowerState
Get all private desktops that are turned on, or are turning on (assuming unmanaged desktops are powered on).

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerPrivateDesktop -Tag TestTag
Retrieve all private desktops tagged with the 'TestTag' tag.
Get-BrokerRebootCycle

Oct 10, 2013

Gets one or more reboot cycles.

Syntax

Get-BrokerRebootCycle -Uid <Int64> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

The Get-BrokerRebootCycle cmdlet is used to enumerate reboot cycles that match all of the supplied criteria.

See about_Broker_Filtering for information about advanced filtering options.

-------------------------- BrokerRebootCycle Object

The reboot cycle object returned represents a single occurrence of the process of rebooting a portion (or all) of the machines in a desktop group.

-- CatalogName (System.String)

Name of the catalog whose machines are rebooted by this cycle if the cycle is associated with a catalog.

-- CatalogUid (System.Int32?)

Uid of the catalog whose machines are rebooted by this cycle if the cycle is associated with a catalog.

-- DesktopGroupName (System.String)

Name of the desktop group whose machines are rebooted by this cycle.

-- DesktopGroupUid (System.Int32)

Uid of the desktop group whose machines are rebooted by this cycle.

-- EndTime (System.DateTime?)

Time at which this cycle was completed, canceled or abandoned.

-- MachinesCompleted (System.Int32)

Number of machines successfully rebooted by this cycle.

-- MachinesFailed (System.Int32)

Number of machines issued with reboot requests where either the request failed or the operation did not complete within
the allowed time.

-- MachinesInProgress (System.Int32)
Number of machines issued with reboot requests but which have not yet completed the operation.

-- MachinesPending (System.Int32)
Number of outstanding machines to be rebooted during the cycle but on which processing has not yet started.

-- MachinesSkipped (System.Int32)
Number of machines scheduled for reboot during the cycle but which were not processed either because the cycle was canceled or abandoned or because the machine was unavailable for reboot processing throughout the cycle.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
Map of metadata associated with this cycle.

-- RebootDuration (System.Int32)
Approximate maximum number of minutes over which the reboot cycle runs.

-- StartTime (System.DateTime)
Time of day at which this reboot cycle was started.

-- State (Citrix.BrokerAdmin.SDK.RebootCycleState)
The execution state of this cycle.

-- Uid (System.Int64)
Unique ID of this reboot cycle.

-- WarningDuration (System.Int32)
Number of minutes to display the warning message for.

-- WarningMessage (System.String)
Warning message to display to users in active sessions prior to rebooting the machine.

-- WarningTitle (System.String)
Title of the warning message dialog.

Related topics
Start-BrokerRebootCycle
Stop-BrokerRebootCycle

Parameters
-Uid<Int64>
Gets reboot cycles that have the specified Uid.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>true</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **CatalogName**<String>

Gets reboot cycles that relate to the named catalog.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **CatalogUid**<Int32>

Gets reboot cycles that relate to the catalog with a particular Uid.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **DesktopGroupName**<String>

Gets reboot cycles that relate to the named desktop group.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **DesktopGroupUid**<Int32>

Gets reboot cycles that relate to the desktop group with a particular Uid.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **EndTime**<DateTime>
Gets reboot cycles that have the specified time at which the reboot cycle was completed, canceled or abandoned.

- **MachinesCompleted**<Int32>
  Gets reboot cycles that have the specified count of machines successfully rebooted during the cycle.

- **MachinesFailed**<Int32>
  Gets reboot cycles that have the specified count of machines issued with reboot requests where either the request failed or the operation did not complete within the allowed time.

- **MachinesInProgress**<Int32>
  Gets reboot cycles that have the specified count of machines issued with reboot requests but which have not yet completed the operation.

- **MachinesPending**<Int32>
  Gets reboot cycles that have the specified count of outstanding machines to be rebooted during the cycle but on which processing has not yet started.
-**MachinesSkipped**<Int32>

Gets reboot cycles that have the specified count of machines scheduled for reboot during the cycle but which were not processed either because the cycle was canceled or abandoned or because the machine was unavailable for reboot processing throughout the cycle.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Metadata**<String>

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**RebootDuration**<Int32>

Gets reboot cycles that have the specified approximate maximum duration in minutes over which the reboot cycle runs.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**StartTime**<DateTime>

Gets reboot cycles that have the specified time at which the reboot cycle started.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**State**<RebootCycleState>

Gets reboot cycles that have the specified overall state of the reboot cycle. Valid values are Initializing, Active, Completed, Canceled, and Abandoned.
-**ReturnTotalRecordCount** `<SwitchParameter>`

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

-**MaxRecordCount** `<Int32>`

Specifies the maximum number of records to return.

-**Skip** `<Int32>`

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-**SortBy** `<String>`

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.
Accept Pipeline Input? | false

-Filter<String>
Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Required? | false
Default Value
Accept Pipeline Input? | false

-Property<String[]>
Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Required? | false
Default Value
Accept Pipeline Input? | false

-AdminAddress<String>
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

Required? | false
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
Accept Pipeline Input? | false

Input Type
None Input cannot be piped to this cmdlet.

Return Values
Citrix.BrokerAdmin.SDK.RebootCycle

Returns matching reboot cycles.

Examples
-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerRebootCycle
Enumerate all reboot cycles.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-BrokerRebootCycle -State Completed
Enumerates all reboot cycles that have successfully completed.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Get-BrokerRebootCycle -DesktopGroupName CallCenter
Enumerates all reboot cycles related to the desktop group named CallCenter.
Get-BrokerRebootSchedule

Oct 10, 2013
Gets one or more reboot schedules.

Syntax

Get-BrokerRebootSchedule [-DesktopGroupUid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

The Get-BrokerRebootSchedule cmdlet is used to enumerate desktop group reboot schedules that match all of the supplied criteria.

A reboot schedule can be configured to cause all of the machines in a desktop group to be rebooted at a particular time each day or each week, with the reboot of the individual machines spread out over the duration of the whole reboot cycle. A specific warning message can be configured to be displayed to users who are running sessions on the machines being rebooted. Note that each desktop group can only have a single reboot schedule configured.

See about_Broker_Filtering for information about advanced filtering options.

-------------------------- BrokerRebootSchedule Object

The reboot schedule object returned represents a regularly scheduled reboot of machines in a desktop group.

--- Active (System.Boolean)

True if there is an active reboot cycle for this schedule, false otherwise.

--- Day (Citrix.Broker.Admin.SDK.RebootScheduleDays)

When the frequency is weekly, day of the week on which the schedule reboot starts.

--- DesktopGroupName (System.String)

Name of the desktop group rebooted by this schedule.

--- DesktopGroupUid (System.Int32)

Uid of the desktop group rebooted by this schedule.

--- Enabled (System.Boolean)

True if this schedule is currently enabled, false otherwise.

--- Frequency (Citrix.BrokerAdmin.SDK.RebootScheduleFrequency)
Whether the schedule runs daily or weekly.

-- RebootDuration (System.Int32)
Approximate maximum number of minutes over which the scheduled reboot cycle runs.

-- StartTime (System.TimeSpan)
Time of day at which the scheduled reboot cycle starts.

-- WarningDuration (System.Int32)
Number of minutes to display the warning message for.

-- WarningMessage (System.String)
Warning message to display to users in active sessions prior to rebooting the machine.

-- WarningTitle (System.String)
Title of the warning message dialog.

Related topics
Set-BrokerRebootSchedule
New-BrokerRebootSchedule
Remove-BrokerRebootSchedule
Get-BrokerRebootCycle

Parameters
-DesktopGroupUid<Int32>
Gets the reboot schedule for the desktop group having this Uid.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-DesktopGroupName<String>
Gets the reboot schedule for the desktop group having this name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
-**Active**<Boolean>

Gets desktop group reboot schedules according to whether they are currently active or not. A schedule is active if there is a reboot cycle currently running that was started as a result of the schedule.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Day**<RebootScheduleDays>

Gets the reboot schedules set to run on the specified day (one of Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Enabled**<Boolean>

Gets the reboot schedules with the specified Enabled value.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Frequency**<RebootScheduleFrequency>

Gets the reboot schedules with the specified frequency (either Weekly or Daily).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**RebootDuration**<Int32>

Gets the reboot schedules with the specified duration.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
## Accept Pipeline Input?

| Accept Pipeline Input? | false |

## -StartTime

**Start Time**<TimeSpan>

Gets the reboot schedules with the specified start time (HH:MM).

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

## -ReturnTotalRecordCount

**Return Total Record Count**<SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

## -MaxRecordCount

**Max Record Count**<Int32>

Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |
| Accept Pipeline Input? | false |

## -Skip

**Skip**<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | false |

## -SortBy

**Sort By**<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.999
no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter</String>**

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Property</String[]>**

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress</String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None Input cannot be piped to this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.RebootSchedule

Returns matching reboot schedules.

Examples
-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerRebootSchedule
Enumerates all of the reboot schedules.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerRebootSchedule -Enabled $false -Frequency Daily
Enumerates all disabled reboot schedules that are scheduled to run daily.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Get-BrokerRebootSchedule -DesktopGroupUid 11
Returns the unique reboot schedule for the desktop group having the Uid 11.
Get-BrokerRemotePCAccount

Oct 10, 2013
Get RemotePCAccount entries configured for this site.

Syntax

Get-BrokerRemotePCAccount -Uid <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Retrieves RemotePCAccounts matching the specified criteria. If no parameters are specified this cmdlet enumerates all RemotePCAccounts. Each RemotePCAccount object defines a set of machines either by machine name patterns or by where the machines are placed in Active Directory, and which RemotePC catalog the machines are to be associated with when they are discovered.

-------------------------- BrokerRemotePCAccount Object

RemotePCAccounts define a set of machines either by machine name patterns or by where the machines are placed in Active Directory, and which RemotePC catalog the machines are to be associated with when they are discovered.

-- AllowSubfolderMatches (System.Boolean)

Specifies whether machines subfolders of specified AD OUs are to be considered part of the RemotePCAccount.

-- CatalogUid (System.Int32)

The Uid of the RemotePC catalog to which machines in the RemotePCAccount automatically join during registration.

-- MachinesExcluded (System.String[])

A list of machines which are to be excluded from the RemotePCAccount. Wildcard matching is supported.

-- MachinesIncluded (System.String[])

A list of machines which are to be included in the RemotePCAccount. Wildcard matching is supported.

-- OU (System.String)

Machines within this specified AD OU are considered part of the RemotePCAccount, unless they are in they match the MachinesExcluded

-- Uid (System.Int32)

The Uid of the RemotePCAccount object.

Related topics

New-BrokerRemotePCAccount
Set-BrokerRemotePCAccount

Remove-BrokerRemotePCAccount

Parameters

- **Uid**<Int32>

  Gets the RemotePCAccount with the specified unique ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **AllowSubfolderMatches**<Boolean>

  Gets RemotePCAccounts with the specified value of AllowSubfolderMatches.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **CatalogUid**<Int32>

  Gets RemotePCAccounts belonging to the specified Remote PC catalog.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **OU**<String>

  Gets the RemotePCAccount with the specified OU.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ReturnTotalRecordCount**<SwitchParameter>

  When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.
-MaxRecordCount\<Int32>\>

Specifies the maximum number of records to return.

-Filter\<String>\>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

-Skip\<Int32>\>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy\<String>\>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.
-**Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.RemotePCAccount

Get-BrokerRemotePCAccount returns an object for each matching RemotePCAccount.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerRemotePCAccount
Find all RemotePCAccounts.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerRemotePCAccount -CatalogUid 42
Find RemotePCAccounts belonging to Remote PC catalog 42.
Get-BrokerResource

Oct 10, 2013
Gets resources that a user can broker connections to.

Syntax


Detailed Description

Retrieve a list of resources that a user has access to, taking into account the site access policy, configuration of desktop groups, assignments, entitlements, and applications.

What a user has access to depends on a number of attributes:

-- User's name or security identifier.
-- Groups that the user is a member of (names or security identifiers).
-- IP address of the client the user connects from.
-- Name of the client that the user connects from.
-- Whether the user is connecting via Citrix Access Gateway.
-- SmartAccess tags when connecting via Citrix Access Gateway.

You must always specify the user's name or security identifier, but you will not always be able to predict what some of the other values will be. By omitting these values the corresponding access checks are ignored.

Consider for example, a site configuration that uses IP address ranges to allow access to private desktop A when connecting from the local network and private desktop B when connecting from home. Running this cmdlet without specifying a client IP address would return both A and B.

The output of this cmdlet depends on the available resources:

-- Assigned private desktops are returned as PrivateDesktop objects.
-- Shared desktops are returned as EntitlementPolicyRule objects.
-- Assign-On-First-Use desktops that have not been assigned yet are returned as AssignmentPolicyRule objects.
-- Application resources produce Application objects.

If more than one type of resource is available, the output pipeline contains a mixture of the above objects, in no particular order.

Only resources accessible based on the specified parameters, and visible to the administrator running this cmdlet are returned.

Related topics
Parameters

- **User**<String>
  
  Gets resources given the specified user name or security identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Groups**<String[]>

  Get resources accessible given a list of group names or security identifiers.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ClientName**<String>

  Get resources given the specified client name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ClientIP**<String>

  Get resources given the specified client IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ViaAG**<Boolean>

  Gets resources given the specified ViaAG setting.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
**Accept Pipeline Input?**
false

**-SmartAccessTags<String[]>*

Get resources given the specified SmartAccess tags.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress<String>*

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**
None You cannot pipe input into this cmdlet.

**Return Values**
Citrix.BrokerAdmin.SDK.PrivateDesktop
Get-BrokerResource returns a PrivateDesktop for each accessible assigned private desktop.
Citrix.BrokerAdmin.SDK.EntitlementPolicyRule
Get-BrokerResource returns an EntitlementPolicyRule object for each accessible entitlement to a shared desktop.
Citrix.BrokerAdmin.SDK.AssignmentPolicyRule
Citrix.BrokerAdmin.SDK.Application
Get-BrokerResource returns an Application object for each accessible application.

**Examples**

-------------------------- EXAMPLE 1 --------------------------
Get-BrokerResource -User MYDOMAIN\User1 -Group MYDOMAIN\Accounts,MYDOMAIN\Managers
List resources visible by User1 assuming membership of a couple of groups.

-------------------------- EXAMPLE 2 --------------------------
[int[]]$groups = (Get-BrokerResource -User MYDOMAIN\User1 | %{ $_.DesktopGroupUid })
Get-BrokerDesktopGroup -Filter { Uid -in $groups } -Property Uid,Name
Get all of the desktop groups supporting the resources accessible by User1, outputting the uid and name of each desktop group.
Get-BrokerScopedObject

Oct 10, 2013
Gets the details of the scoped objects for the Broker Service.

Syntax

Get-BrokerScopedObject -ScopeId <Guid> [-Property <String[]>] [-AdminAddress <String>] [CommonParameters]


Detailed Description

Returns a list of directly scoped objects including the names and identifiers of both the scope and object as well as the object description for display purposes.

There will be at least one result for every directly scoped object. When an object is associated with multiple scopes the output contains one result per scope duplicating the object details.

No records are returned for the All scope, though if an object is not in any scope a result with a null ScopeId and ScopeName will be returned.

-------------------------- BrokerScopedObject Object

A scoped, or scopeable object configured in the broker.

-- Description (System.String)

Description of the object (possibly $null if the object type does not have a description).

-- ObjectId (System.String)

Unique identifier of the object.

-- ObjectName (System.String)

Display name of the object

-- ObjectType (Citrix.Broker.Admin.SDK.ScopedObjectType)

Type of the object this entry relates to.

-- ScopeId (System.Guid?)

Specifies the unique identifier of the scope.

-- ScopeName (System.String)

Specifies the display name of the scope.

Related topics
Parameters

- **ScopeId**: Guid
  
  Gets scoped object entries for the given scope identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **Description**: String
  
  Gets scoped object entries for objects with the specified description.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ObjectId**: String
  
  Gets scoped object entries for objects with the specified object identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ObjectName**: String
  
  Gets scoped object entries for objects with the specified object identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ObjectType**: ScopedObjectType
  
  Gets scoped object entries for objects of the given type.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | false  
--- | ---  
**-ScopeName**<String>  
Gets scoped object entries with the given scope name.  

| Required? | false  
| --- | ---  
| Default Value |  
| Accept Pipeline Input? | false  

**-ReturnTotalRecordCount**<SwitchParameter>  
When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.  

| Required? | false  
| --- | ---  
| Default Value | False  
| Accept Pipeline Input? | false  

**-MaxRecordCount**<Int32>  
Specifies the maximum number of records to return.  

| Required? | false  
| --- | ---  
| Default Value | 250  
| Accept Pipeline Input? | false  

**-Skip**<Int32>  
Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.  

| Required? | false  
| --- | ---  
| Default Value | 0  
| Accept Pipeline Input? | false  

**-SortBy**<String>  
Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.
-Filter <String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

-Property <String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

Return Values

Citrix.BrokerSdk.ScopedObject

The Get-BrokerScopedObject command returns an object containing the following properties:

ScopeId <Guid?>

Specifies the unique identifier of the scope.

ScopeName <String>

Specifies the display name of the scope.
ObjectType <ScopedObjectType>
Type of the object this entry relates to.

ObjectId <String>
Unique identifier of the object.

ObjectName <String>
Display name of the object

Description <String>
Description of the object (possibly $null if the object type does not have a description).

Notes
If the command fails, the following errors can be returned.

Error Codes
-------------
PartialData
Only a subset of the available data was returned.

InvalidFilter
A filtering expression was included that could not be interpreted for this command.

PermissionDenied
You do not have permission to execute this command.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-BrokerScopedObject -ObjectType Catalog

    ScopeId : eff6f464-f1ee-4442-add3-99982e0cec01
    ScopeName : Sales
    ObjectType : Catalog
    ObjectId : 1
    ObjectName : MyExampleCatalog
Description : Test Catalog

ScopeId : 304e0fa7-d390-47f0-a94f-7e956a324c41
ScopeName : Finance
ObjectType : Catalog
ObjectId : 1
ObjectName : MyExampleCatalog
Description : Test Catalog

ScopeId :
ScopeName :
ObjectType : Catalog
ObjectId : 2
ObjectName : AnotherCatalog
Description : Another catalog in no scopes

Gets all of the scoped objects with type Catalog. The example output shows a catalog object (MyExampleCatalog) in two scopes Sales and Finance, and another catalog (AnotherCatalog) that is not in any scope. The ScopeId and ScopeName values returned are null in the final record.
Get-BrokerServiceAddedCapability

Oct 10, 2013
Gets any added capabilities for the Broker Service on the controller.

Syntax
Get-BrokerServiceAddedCapability [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Enables updates to the Broker Service on the controller to be detected. There is no requirement for a database connection to be configured in order for this command to be used.

Related topics
Parameters
- **-AdminAddress** <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type
None You cannot pipe input into this cmdlet.

Return Values

Examples
-------------- EXAMPLE 1 --------------
C:\PS> Get-BrokerServiceAddedCapability
Gets the added capabilities of the Broker Service.
Get-BrokerServiceInstance

Oct 10, 2013
Gets the service instance entries for the Broker Service.

Syntax

Get-BrokerServiceInstance [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

This returns the service interfaces published by instances of the Broker Service. These instances can be used to register the Service with a Configuration Service so that other services are able to utilize their functionality. There is no requirement to have configured a database connection for the Service to use this command.

Related topics

Get-BrokerServiceStatus
Reset-BrokerServiceGroupMembership

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.BrokerAdmin.SDK.ServiceInstance

Get-BrokerServiceInstance returns an opaque object containing information about the Broker WCF Service.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerServiceInstance
Gets all the service instances of the Broker service running on the connected server. (If AdminAddress has not been set for the runspace, this is the Service running on the local machine.) To get the service instances of remote services, use the AdminAddress parameter. This defines the service required by the interfaces.
Get-BrokerServiceStatus

Oct 10, 2013
Determines the current state of the Broker Service on the controller.

Syntax

Get-BrokerServiceStatus [-AdminAddress <String>] [CommonParameters]

Detailed Description

This command provides the ability for the status of the Broker Service in the controller to be determined. Before using this command, you don't have to configure the database connection to the Service.

Related topics

Get-BrokerServiceInstance
Reset-BrokerServiceGroupMembership

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

System.String

Get-BrokerServiceStatus returns a string describing the status of the Broker Service.

DBUnconfigured

The broker does not have a database connection configured.

DBRejectedConnection

The database rejected the logon from the Broker Service. This may be caused by bad credentials, or the database not being installed.

InvalidDBConfigured
The database schema is missing (possibly just the stored procedures in it).

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBMissingOptionalFeature**

The broker is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

**DBMissingMandatoryFeature**

The broker is connected to a database that is valid, but it does not have the full functionality required so the broker cannot function. Upgrading the database is required.

**DBNewerVersionThanService**

The broker is too old to use the database. A newer version is required.

**DBOlderVersionThanService**

The database is too old for the Broker Service. Upgrade the database.

**DBVersionChangeInProgress**

A database schema upgrade is in progress.

**OK**

The broker is connected to a database that is valid, and the service is running.

**PendingFailure**

Connectivity between the Broker Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

**Failed**

Connectivity between the broker and the database has been lost for an extended period of time, or has failed due to a configuration problem. The broker service cannot operate while its connection to the database is unavailable.

**Unknown**

The Service's status cannot be determined.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerServiceStatus
Get the current status of the Service.
Get-BrokerSession

Oct 10, 2013
Gets a list of sessions.

Syntax

Get-BrokerSession -Uid <Int64> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Retrieves sessions matching all the specified criteria. If no parameters are specified this cmdlet enumerates all sessions.

--------------------------- BrokerSession Object

The session object returned represents a session on a machine in the site. The session could be for a desktop or application

-- AgentVersion (System.String)

Version of the Citrix Virtual Delivery Agent (VDA) installed on the machine.

-- ApplicationsInUse (System.String[])

List of SDK Name property of applications in use in the session.

-- AutonomouslyBrokered (System.Boolean)

Session property indicating if the session was started without the use of the broker.

-- BrokeringDuration (System.Int32?)

Time taken to broker the session.

-- BrokeringTime (System.DateTime?)
Time at which the session was brokered.

-- BrokeringUserName (System.String)
The user name of the brokering user.

-- BrokeringUserSID (System.String)
The SID of the brokering user.

-- CatalogName (System.String)
The name of the catalog that the machine hosting the session is assigned to.

-- ClientAddress (System.String)
The IP address of the client connected to the desktop.

-- ClientName (System.String)
The host name of the client connected to the session.

-- ClientVersion (System.String)
The version of the Citrix Receiver running on the client connected to the session.

-- ConnectedViaHostName (System.String)
The host name of the incoming connection. This is usually a gateway, router or client.

-- ConnectedViaIP (System.String)
The IP address of the incoming connection This is usually a gateway, router or client.

-- ControllerDNSName (System.String)
The DNS host name of the controller that the session's hosting machine is registered with.

-- DesktopGroupName (System.String)
Name of the desktop group of the machine the session is on.

-- DesktopGroupUid (System.Int32)
UID of the desktop group of the machine the session is on.

-- DesktopKind (Citrix.BrokerAdmin.SDK.DesktopKind)
Indicates if the session is shared or private.

-- DesktopSID (System.String)
The Windows SID of the machine the session is on.

-- DesktopUid (System.Int32)
For a desktop session, the unique identifier of the desktop.

-- DeviceId (System.String)
Unique identifier for the client device that has most recently been associated with the session.

-- DNSName (System.String)
The DNS host name of the machine hosting the session.

-- EstablishmentDuration (System.Int32?)
Duration that it took to establish the session.

-- EstablishmentTime (System.DateTime?)
Time at which the session was established.

-- HardwareId (System.String)
Unique identifier for the client hardware that has been most recently associated with the session.

-- Hidden (System.Boolean)
Flag to indicate if the session is currently hidden from the user and not to be reconnected to.

-- HostedMachineName (System.String)
The friendly name of a hosted machine running the session, as used by its hypervisor. This does not necessarily match either the DNS or AD name of the machine.

-- HostingServerName (System.String)
DNS name of the hypervisor that is hosting the machine hosting the session.

-- HypervisorConnectionName (System.String)
The name of the hypervisor connection that the machine hosting the session has been assigned to.

-- ImageOutOfDate (System.Boolean?)
Denotes whether the VM image for a hosted machine is out of date and due to be updated to a new master image when the machine next reboots.

-- InMaintenanceMode (System.Boolean)
Denotes whether the machine hosting the session is in maintenance mode.

-- IPAddress (System.String)
The IP address of the machine hosting the session.

-- IsPhysical (System.Boolean)
This value is false if the machine hosting the session can be power managed, and true otherwise
-- LaunchedViaHostName (System.String)
The host name of the StoreFront server used to launch the session.

-- LaunchedViaIP (System.String)
The IP address of the StoreFront server used to launch the session.

-- MachineName (System.String)
DNS host name of the machine hosting the session.

-- MachineSummaryState (Citrix.BrokerAdmin.SDK.DesktopSummaryState)
The summary state of the machine (will be Unregistered, Disconnected, or InUse)

-- MachineUid (System.Int32)
UID of the machine hosting the session.

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
Map of metadata for this session.

-- OSType (System.String)
A string that can be used to identify the operating system that is running on the machine hosting the session.

-- PersistUserChanges (Citrix.BrokerAdmin.SDK.PersistUserChanges)
Describes whether/how the user changes are persisted. Possible values are:
  o OnLocal - Persist the user changes locally.
  o Discard - Discard user changes.
  o OnPvd - Persist user changes on the Citrix Personal vDisk.

-- PowerState (Citrix.BrokerAdmin.SDK.PowerState)
The current power state of the machine hosting the session. Possible values are: Unmanaged, Unknown, Unavailable, On, Suspended, TurningOn, TurningOff, Suspending and Resuming.

-- Protocol (System.String)
The protocol that the session is using. It can be either "HDX" or "RDP".

-- ProvisioningType (Citrix.BrokerAdmin.SDK.ProvisioningType)
Describes how the machine hosting the session was provisioned, possible values are:
  o Manual: No provisioning.
  o PVS: Machine provisioned by PVS (may be physical, blade, VM,...)
  o MCS: Machine provisioned by MCS (machine must be VM)
-- SecureIcaActive (System.Boolean?)

Indicates whether SecureICA is active on the session.

-- SessionId (System.Int32)

Deprecated. A unique identifier that Remote Desktop Services uses to track the session but it is only unique on that machine.

-- SessionKey (System.Guid)

GUID that provides a unique identifier for this session.

-- SessionState (Citrix.BrokerAdmin.SDK.SessionState)

The state of the session. Valid values are PreparingNewSession, Connected, Active, Disconnected, Reconnecting, NonBrokeredSession, Other, and Unknown.

-- SessionStateChangeTime (System.DateTime)

The time of the most recent state change for the session.

-- SessionSupport (Citrix.BrokerAdmin.SDK.SessionSupport)

Indicates if the machine hosting the session supports multiple or single sessions.

-- SessionType (Citrix.BrokerAdmin.SDK.SessionType)

Indicates if this is an Application or Desktop session.

-- SmartAccessTags (System.String[])

The Smart Access tags for this session.

-- StartTime (System.DateTime?)

Indicates when the session was started.

-- Uid (System.Int64)

Unique identifier of this session.

-- UserFullName (System.String)

The full name of the user.

-- UserName (System.String)

The name of the user.

-- UserSID (System.String)

The user's Windows SID.

-- UserUPN (System.String)
The user’s User Principal Name

Related topics

Disconnect-BrokerSession

Stop-BrokerSession

Get-BrokerDesktop

Parameters

-\textbf{Uid}<\text{Int64}>

Get session by its Uid.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-\textbf{AgentVersion}<\text{String}>

Gets sessions with a specific Virtual Desktop Agent version.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-\textbf{ApplicationInUse}<\text{String}>

Gets sessions running specific applications (identified by their SDK Name property).

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-\textbf{AutonomouslyBrokered}<\text{Boolean}>

Get autonomously brokered sessions.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |
**-BrokeringDuration<Int32>**

Gets session with a specific time taken to broker. In general, Citrix recommends using -Filter and relative comparisons.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-BrokeringTime<DateTime>**

Get sessions brokered at a specific time. In general, Citrix recommends using -Filter and relative comparisons.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-BrokeringUserName<String>**

Get sessions by brokering user.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-BrokeringUserSID<String>**

Get sessions by brokering user SID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-CatalogName<String>**

Gets sessions on machines from a specific catalog name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**-ClientAddress**<String>

Get sessions by client IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ClientName**<String>

Get sessions by client name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ClientVersion**<String>

Get sessions by client version.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ConnectedViaHostName**<String>

Get sessions by host name of the incoming connection. This is usually a proxy or Citrix Access Gateway server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ConnectedViaIP**<String>

Get sessions by IP address of the incoming connection.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
### -ControllerDNSName<String>

Gets sessions that are hosted on machines which are registered with a specific controller.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -DesktopGroupName<String>

Gets sessions from a desktop group with the specified name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -DesktopGroupId<Int32>

Gets sessions from a desktop group with the specified UID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -DesktopKind<DesktopKind>

Gets sessions on a desktop of a particular kind.

Valid values are Private and Shared.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -DesktopSID<String>

Get sessions by desktop SID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

### Accept Pipeline Input?

| Accept Pipeline Input? | false |

<table>
<thead>
<tr>
<th><strong>-DesktopUid</strong>&lt;Int32&gt;</th>
</tr>
</thead>
</table>

**Get sessions by desktop Uid.**

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

<table>
<thead>
<tr>
<th><strong>-DeviceId</strong>&lt;String&gt;</th>
</tr>
</thead>
</table>

**Get sessions by client device id.**

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

<table>
<thead>
<tr>
<th><strong>-DNSName</strong>&lt;String&gt;</th>
</tr>
</thead>
</table>

**Gets sessions by their machine's DNS name.**

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

<table>
<thead>
<tr>
<th><strong>-EstablishmentDuration</strong>&lt;Int32&gt;</th>
</tr>
</thead>
</table>

**Gets sessions which took a specific time to establish. In general, Citrix recommends using -Filter and relative comparisons.**

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

<table>
<thead>
<tr>
<th><strong>-EstablishmentTime</strong>&lt;DateTime&gt;</th>
</tr>
</thead>
</table>

**Gets sessions which became established at a particular time. In general, Citrix recommends using -Filter and relative comparisons.**

<p>| Required? | false |
| Default Value |</p>
<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>false</th>
</tr>
</thead>
</table>

- **HardwareId**: String

Get sessions by client hardware id.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Hidden**: Boolean

Get sessions by whether they are hidden or not. Hidden sessions are treated as though they do not exist when brokering sessions; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **HostedMachineName**: String

Gets sessions by their machine's name as known to its hypervisor.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **HostingServerName**: String

Gets sessions hosted by a machine with a specific name of the hosting hypervisor server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **HypervisorConnectionName**: String

Gets sessions hosted by a machine with a specific name of the hosting hypervisor connection.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
</tr>
<tr>
<td>ImageOutOfDate</td>
<td>Boolean</td>
</tr>
<tr>
<td>Required?</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
<tr>
<td>InMaintenanceMode</td>
<td>Boolean</td>
</tr>
<tr>
<td>Required?</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
<tr>
<td>IPAddress</td>
<td>String</td>
</tr>
<tr>
<td>Required?</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
<tr>
<td>IsPhysical</td>
<td>Boolean</td>
</tr>
<tr>
<td>Required?</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
<tr>
<td>LaunchedViaHostName</td>
<td>String</td>
</tr>
</tbody>
</table>
### LaunchedViaIP

Get sessions by the IP address of the Web Interface server from which a user launches a session.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### MachineName

Gets sessions by their machine name (DOMAIN\MACHINE).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### MachineSummaryState

Gets sessions on a machine with a specific summary state. Valid values are Off, Unregistered, Available, Disconnected, Preparing, and InUse.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### MachineUid

Gets sessions on a machine with the specified UID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### Metadata

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**OSType**<String>

Gets sessions with a specific type of operating system.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**PersistUserChanges**<PersistUserChanges>

Gets sessions where the user changes are persisted in a particular manner. Values can be:

- OnLocal - User changes are persisted locally.
- Discard - User changes are discarded.
- OnPvd - User changes are persisted on the Pvd.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**PowerState**<PowerState>

Gets sessions on machines in the specified power state.

Valid values are Unmanaged, Unknown, Unavailable, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Protocol**<String>
Get sessions by connection protocol. Valid values are HDX and RDP.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ProvisioningType**

Gets sessions hosted on machines provisioned in a particular manner. Values can be:

- Manual - No automated provisioning.
- PVS - Machine provisioned by PVS (machine may be physical, blade, VM,...).
- MCS - Machine provisioned by MCS (machine must be VM).

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **SecureIcaActive**

Get sessions by their use of SecureICA.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **SessionId**

Deprecated.

Gets sessions by session ID, a unique identifier that Remote Desktop Services uses to track the session but it is only unique on that machine.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **SessionKey**

Gets session having the specified unique key.
- **SessionState** `<SessionState>`
  
  Get sessions by their state.
  
  Valid values are Other, PreparingNewSession, Connected, Active, Disconnected, Reconnecting, NonBrokeredSession, and Unknown.

- **SessionStateChangeTime** `<DateTime>`
  
  Get sessions by their last state change time. In general, Citrix recommends using -Filter and relative comparisons.

- **SessionSupport** `<SessionSupport>`
  
  Gets sessions hosted on machines which support the required pattern of sessions. Values can be:
  
  - SingleSession - Single-session only machine.
  - MultiSession - Multi-session capable machine.

- **SessionType** `<SessionType>`
  
  Get sessions by their type.
  
  Valid values are Application and Desktop.
### -StartTime

Get sessions by their start time. In general, Citrix recommends using -Filter and relative comparisons.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -UserFullName

Gets sessions by user’s full name (usually ‘first-name last-name’).

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -UserName

Get sessions by user name (in the form domain\user).

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -UserSID

Get sessions by user’s Windows SID.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -UserUPN

Gets sessions by user’s User Principal Name (in the form user@domain).

| Required? | false |
### ApplicationUid (Int32)

Get sessions running the application with the specified Uid.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### SharedDesktopUid (Int32)

Get sessions by SharedDesktop Uid.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### ReturnTotalRecordCount (SwitchParameter)

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

### MaxRecordCount (Int32)

Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |
| Accept Pipeline Input? | false |

### Skip (Int32)

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.
### -SortBy <String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -Filter <String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -Property <String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.Session

Returns sessions matching the specified criteria.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerSession
Enumerate all sessions.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerSession -UserName MyDomain\MyAccount -SessionState Disconnected
Get all disconnected sessions for a specific user.
Get-BrokerSharedDesktop

Oct 10, 2013
Get shared desktops configured for this site.

Syntax

Get-BrokerSharedDesktop [-Uid] <Int32> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

This cmdlet is deprecated, please use the Get-BrokerMachine cmdlet instead.

Retrieve shared desktops matching the specified criteria. If no parameters are specified, all shared desktops are enumerated.

Get-BrokerSharedDesktop returns configuration information only for shared desktops (a DesktopKind of 'Shared').

For information about advanced filtering options, see about_Broker_Filtering; for information about desktops, see about_Broker_Desktops.

-------------------------- BrokerSharedDesktop Object

Shared desktops are desktops that are assigned randomly to users upon connection from a pool of available machines.

-- AgentVersion (System.String)
Version of the Citrix Virtual Delivery Agent (VDA) installed on the desktop.

-- ControllerDNSName (System.String)
The DNS host name of the controller that the desktop is registered to.

-- DesktopGroupUid (System.Int32)
Uid of the desktop group the desktop has been assigned to.

-- DNSName (System.String)
The DNS host name of the desktop.

-- HostedMachineld (System.String)
Unique ID within the hosting unit of the target managed desktop.
The friendly name of a hosted desktop as used by its hypervisor. This is not necessarily the DNS name of the desktop.

-- HostedMachineName (System.String)

The friendly name of a hosted desktop as used by its hypervisor. This is not necessarily the DNS name of the desktop.

-- HostingServerName (System.String)

DNS name of the hypervisor that is hosting the desktop if managed.

-- HypervisorConnectionUid (System.Int32?)

The UID of the hypervisor connection that the desktop has been assigned to, if managed.

-- InMaintenanceMode (System.Boolean)

Denotes whether the desktop is in maintenance mode.

-- IPAddress (System.String)

The IP address of the desktop.

-- LastDeregistrationReason (Citrix.BrokerAdmin.SDK.DeregistrationReason?)

The reason for the last deregistration of the desktop with the broker. Possible values are:

AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

-- LastDeregistrationTime (System.DateTime?)

Time of the last deregistration of the desktop from the controller.

-- LastHostingUpdateTime (System.DateTime?)

Time of last update to any hosting data for this desktop reported by the hypervisor connection.

-- MachineName (System.String)

DNS host name of the machine associated with the desktop.

-- OSType (System.String)

A string that can be used to identify the operating system that is running on the desktop.

-- OSVersion (System.String)

A string that can be used to identify the version of the operating system running on the desktop, if known.

-- PowerState (Citrix.BrokerAdmin.SDK.PowerState)

The current power state of the desktop. Possible values are: Unmanaged, Unknown, Unavailable, Off, On, Suspended.
TurningOn, TurningOff, Suspending, resuming.

-- RegistrationState (Citrix.BrokerAdmin.SDK.RegistrationState)
Indicates the registration state of the desktop. Possible values are: Unregistered, Initializing, Registered, AgentError.

-- SID (System.String)
The security identifier of the shared desktop.

-- Uid (System.Int32)
The uid of the shared desktop.

-- WillShutdownAfterUse (System.Boolean)
Flag indicating whether this desktop is tainted and will be shutdown after all sessions on the desktop have ended. This flag should only ever be true on power managed, single-session desktops.

Note: The desktop will not shut down if it is in maintenance mode, but will shut down after the desktop is taken out of maintenance mode.

Related topics
Set-BrokerSharedDesktop

Parameters

- **Uid**:<Int32>
  Gets desktops by Uid.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **MachineName**:<String>
  Gets desktops by machine name (in the form 'domain\machine').

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AgentVersion**:<String>
  Gets desktops with a specific Virtual Desktop Agent version.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
### Default Value

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>false</th>
</tr>
</thead>
</table>

### ControllerDNSName

**Type:** String

**Description:**

Gets desktops by the DNS name of the controller they are registered with.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### DesktopGroupUid

**Type:** Int32

**Description:**

Gets desktops from a desktop group with a specific Uid.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### DNSName

**Type:** String

**Description:**

Gets desktops by DNS name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### HostedMachineId

**Type:** String

**Description:**

Gets desktops by the machine id known to the hypervisor.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### HostedMachineName

**Type:** String

**Description:**

Gets desktops by the machine name known to the hypervisor.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
-HostingServerName<String>

Gets desktops by the name of the hosting hypervisor server.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-HypervisorConnectionUid<Int32>

Gets desktops by the uid of the hosting hypervisor connection.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-InMaintenanceMode<Boolean>

Gets desktops by the InMaintenanceMode setting.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-IPAddress<String>

Gets desktops by IP address.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-LastDeregistrationReason<DeregistrationReason>

Gets desktops whose broker last recorded a specific deregistration reason.

Valid values are $null, AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest,
MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **LastDeregistrationTime**<DateTime>

Gets desktops by the time that they were last deregistered.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **LastHostingUpdateTime**<DateTime>

Gets desktops by the time that the hosting information was last updated.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **OSType**<String>

Gets desktops by the type of operating system they are running.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **OSVersion**<String>

Gets desktops by the version of the operating system they are running.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input?  false

**-PowerState**<PowerState>

Gets desktops by power state.

Valid values are Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-RegistrationState**<RegistrationState>

Gets desktops by registration state.

Valid values are Registered, Unregistered, and AgentError.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SID**<String>

Gets desktops by machine SID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Tag**<String>

Get desktops tagged with the given tag.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-WillShutdownAfterUse**<Boolean>

Gets desktops depending on whether they shutdown after use or not.
**-ReturnTotalRecordCount** `<SwitchParameter>`

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-MaxRecordCount** `<Int32>`

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Skip** `<Int32>`

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SortBy** `<String>`

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AdminAddress

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.SharedDesktop

Get-BrokerSharedDesktop returns an object for each matching shared desktop.

Notes

To compare dates/times, use -Filter and relative comparisons. For more information, see about_Broker_Filtering.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerSharedDesktop -HostingServerName BigServer12*
Get all shared desktops hosted by the hypervisor server BigServer12.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerSharedDesktop -OSType "Windows XP"* | ft -a MachineName,OSType,OSVersion
List all shared desktops running Windows XP, including the machine name and OS details.
Get-BrokerSite

Oct 10, 2013
Gets the current XenDesktop broker site.

Syntax
Get-BrokerSite [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
The Get-BrokerSite cmdlet gets the current broker site.

The broker site is a top-level, logical representation of the XenDesktop site, from the perspective of the brokering services running within the site. It defines various site-wide default attributes used by the brokering services.

A XenDesktop installation has only a single broker site instance.

-------------------------- BrokerSite Object

The BrokerSite object represents logical representation of the XenDesktop site. It contains the following properties:

-- BaseOU (System.Guid?)

The objectGUID property identifying the base OU in Active Directory used for desktop registrations.

-- BrokerServiceGroupUid (System.Guid)

The Uid for the Broker Service Group.

-- ColorDepth (Citrix.Broker.Admin.SDK.ColorDepth)

The default color depth for new desktop groups.

-- ConfigurationServiceGroupUid (System.Guid?)

The Uid for the Configuration Service Group.

-- DesktopGroupIconUid (System.Int32)

The default desktop icon used for new desktop groups.

-- DnsResolutionEnabled (System.Boolean)

The setting to configure whether numeric IP address or the DNS name to be present in the ICA file.

-- LicensedSessionsActive (System.Int32?)

The count of active licensed session.

-- LicenseEdition (System.String)

The license edition for session brokering.
-- LicenseGraceSessionsRemaining (System.Int32?)
The count of License Grace Session Remaining

-- LicenseModel (Citrix.Broker.Admin.SDK.LicenseModel?)
The licensing model in use. Values can be 'Concurrent' or 'UserDevice'

-- LicenseServerName (System.String)
The DNS for License Server Name

-- LicenseServerPort (System.Int32)
The port for the License Server

-- LicensingBurnInDate (System.DateTime?)
The date for the license to end

-- LicensingGraceHoursLeft (System.Int32?)
The number of grace hours left after license expiry

-- LicensingGracePeriodActive (System.Boolean?)
The indicator for licensing grace period active

-- LicensingOutOfBoxGracePeriodActive (System.Boolean?)
The indicator for licensing out of the box grace period active

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
The metadata for this command.

-- Name (System.String)
The name of the site

-- SecureIcaRequired (System.Boolean)
The default SecureICA usage requirements for new desktop groups.

-- TrustRequestsSentToTheXmlServicePort (System.Boolean)
The XML Service trust settings.

Related topics

Set-BrokerSite

Get-BrokerIcon

Parameters

-AdminAddress<String>
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.Site

Get-BrokerSite returns the single broker site instance.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerSite
Gets the current broker site.
Get-BrokerTag

Oct 10, 2013
Gets one or more tags.

Syntax

Get-BrokerTag [-Uid] <Int32> [-Property <String>[]] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Gets tags that match all of the supplied criteria.

----------------------- BrokerTag Object

The BrokerTag object represents a single instance of a Tag associated to other objects. It contains the following properties:

-- MetadataMap (System.Collections.Generic.Dictionary<string, string>)
The metadata for this command.

-- Name (System.String)
The name of the Tag

-- Uid (System.Int32)
The Uid of the Tag

-- UUID (System.Guid)
The UUID associated to the Tag

Related topics

Add-BrokerTag
New-BrokerTag
Remove-BrokerTag
Rename-BrokerTag

Parameters

-Uid <Int32>
Gets the tag identified by Uid
<table>
<thead>
<tr>
<th>Field</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong>&lt;String&gt;</td>
<td>true</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Gets tags that match the specified name.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| **Metadata**<String> | false     |           |                        |
| Gets records with matching metadata entries.                  |
| The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x". | |
| Required?      | false     |           |                        |
| Default Value  |           |           |                        |
| Accept Pipeline Input? | false |           |                        |

| **UUID**<Guid> | false     |           |                        |
| Gets tags associated with a given UUID.                        |
| Required?      | false     |           |                        |
| Default Value  |           |           |                        |
| Accept Pipeline Input? | false |           |                        |

| **DesktopUid**<Int32> | false     |           |                        |
| Gets tags associated with a Desktop.                          |
| Required?      | false     |           |                        |
| Default Value  |           |           |                        |
| Accept Pipeline Input? | false |           |                        |

| **DesktopGroupUid**<Int32> | false     |           |                        |
|                           |           |           |                        |
Gets tags associated with a desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

**-ReturnTotalRecordCount** `<SwitchParameter>`

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>False</td>
<td>false</td>
</tr>
</tbody>
</table>

**-MaxRecordCount** `<Int32>`

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>250</td>
</tr>
</tbody>
</table>

**-Skip** `<Int32>`

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>0</td>
</tr>
</tbody>
</table>

**-SortBy** `<String>`

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
</table>
The default sort order is by name or unique identifier.

Accept Pipeline Input? | false

-Filter<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Required? | false
Default Value
Accept Pipeline Input? | false

-Property<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Required? | false
Default Value
Accept Pipeline Input? | false

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

Required? | false
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
Accept Pipeline Input? | false

Input Type
None Input cannot be piped to this cmdlet.

Return Values
Citrix.BrokerAdmin.SDK.Tag
Returns matching tags.

Examples
-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerTag
Enumerate all tags

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerTag -Uid 5
Get a single specific tag with a Uid of 5.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Get-BrokerTag -DesktopUid 1
Get tags associated with Desktop 1.
Get-BrokerUnconfiguredMachine

Oct 10, 2013
Gets machines that have registered but are not yet configured in this site.

Syntax

Get-BrokerUnconfiguredMachine -SID <String> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Retrieve machines matching the specified criteria, where the machine has registered with a controller in the site but the machine has not yet been configured to be part of the site. If no parameters are specified, this cmdlet enumerates all such machines.

See about_Broker_Filtering for information about advanced filtering options, and about_Broker_Machines for background information about machines.

--------------------------BrokerUnconfiguredMachine Object

An unconfigured machine is a machine that has registered with the site but is not configured in either a desktop group or a catalog.

-- AgentVersion (System.String)
Version of the Citrix Virtual Delivery Agent (VDA) installed on the unconfigured machine.

-- ControllerDNSName (System.String)
The DNS name of the controller that the unconfigured machine is registered with.

-- DNSName (System.String)
The DNS name of the unconfigured machine.

-- FunctionalLevel (Citrix.Broker.Admin.SDK.FunctionalLevel?)
The functional level of the unconfigured machine. This is determined by the version of the Citrix VDA software installed on the machine.

-- LastDeregistrationTime (System.DateTime?)
The time when the unconfigured machine last deregistered with the Citrix Broker Service.

-- MachineName (System.String)
The machine name of the unconfigured machine in the form domain\machine.

-- OSType (System.String)
The type of operating system installed on the unconfigured machine.

-- OSVersion (System.String)
The version of the operating system installed on the unconfigured machine.

-- SessionSupport (Citrix.Broker.Admin.SDK.SessionSupport?)
The session support of the unconfigured machine. Valid values are:

SingleSession, MultiSession
--- SID (System.String)
Security identifier of the unconfigured machine.

Related topics
Get-BrokerMachine

Parameters
- **SID**:String>

Get machines by their machine SID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **MachineName**:String>

Get machines by their machine name (in the form domain\machine).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **AgentVersion**:String>

Get machines with a specific Virtual Desktop Agent version.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ControllerDNSName**:String>

Get machines by the DNS name of the controller they are registered with.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **DNSName**:String>

Get machines by their DNS name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
-**FunctionalLevel** <FunctionalLevel>

Gets machines with a specific FunctionalLevel.

Valid values are L5, L7

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-**LastDeregistrationTime** <DateTime>

Gets machines by the time that they were last deregistered.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-**OSType** <String>

Gets machines by the type of operating system they are running.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-**OSVersion** <String>

Gets machines by the version of the operating system they are running.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-**SessionSupport** <SessionSupport>

Gets machines that have the specified session capability. Values can be:

- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-**ReturnTotalRecordCount** <SwitchParameter>

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.
### -MaxRecordCount<Int32>

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -Skip<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -SortBy<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -Filter<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -Property<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.UnconfiguredMachine

Get-BrokerUnconfiguredMachine returns an object for each matching machine

Notes

It is generally better to compare dates and times using -Filter and relative comparisons. See about_Broker_Filtering and the examples in this topic for more information.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerUnconfiguredMachine -Filter { ControllerDNSName -eq 'Controller1.domain.com' -and OSTYPE -eq 'Windows XP Service Pack 3' }

This command returns all unconfigured XP SP3 machines which are registered with the controller called 'Controller1.domain.com'.
Get-BrokerUser

Oct 10, 2013
Gets broker users configured for this site.

Syntax

Get-BrokerUser -SID <String> [-Property <String[]>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Retrieve broker users matching the specified criteria. If no parameters are specified this cmdlet enumerates all broker users.

For information about advanced filtering options, see about_Broker_Filtering.

-------------------------- BrokerUser Object

The BrokerUser object represents a single instance of an user. It contains the following properties:

-- FullName (System.String)

The full name of an user

-- Name (System.String)

The name of an user

-- SID (System.String)

The SID of an user

-- UPN (System.String)

The UPN of an user

Related topics

Add-BrokerUser
Remove-BrokerUser

Parameters

-SID <String>

Gets the broker user with the specified SID property value.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
### -Name <String>

Gets the broker user with the specified Name property.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -FullName <String>

Gets the broker user with the specified FullName property.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -UPN <String>

Gets the broker user with the specified UPN property value.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -ApplicationUid <Int32>

Gets broker users associated with the application with the specified Uid.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -MachineUid <Int32>

Gets broker users associated with the broker machine with the specified Uid.

| Required? | false |
| Default Value |
-PrivateDesktopUid:Int32

Gets broker users associated with the private desktop with the specified Uid.

-ReturnTotalRecordCount:SwitchParameter

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

-MaxRecordCount:Int32

Specifies the maximum number of records to return.

-Skip:Int32

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy:String

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if
no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Filter**<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.User

Get-BrokerUser returns an object for each matching broker user.

Examples
-------------------------- EXAMPLE 1 --------------------------
Get-BrokerUser DOMAIN7*  
Get all broker user objects with names matching the supplied pattern

-------------------------- EXAMPLE 2 --------------------------
$pdt = Get-BrokerPrivateDesktop DOMAIN\MACHINENAME
    Get-BrokerUser -PrivateDesktopUid $pdt.Uid
Get all broker user objects added to the specified private desktop
Groups and counts desktops with the same value for a specified property.

**Syntax**

```powershell
Group-BrokerDesktop [-Uid] <Int32> -Property <String> [AdminAddress <String>] [CommonParameters]
```

**Detailed Description**

This cmdlet is now deprecated, please use Group-BrokerMachine.

Filters desktops using the specified criteria, then groups and counts matching desktops with the same value for a particular property. The number of desktops in the group, and the property value for the group, is output. For example:

```
C:\PS> Group-BrokerDesktop -Property SummaryState
Count Name
----- ----
43 Available
17 InUse
3 Disconnected
```

Filtering supports the same options as the Get-BrokerDesktop cmdlet, and allows filtering on both desktop and session properties.

Group-BrokerDesktop is similar to the standard PowerShell Group-Object, but is faster than piping the output of Get-BrokerDesktop into Group-Object when working with many desktops.

Note that all session information properties for multi-session desktops is always $null, this means that is is not possible to group these desktops by session information using this command. Use Get-BrokerSession to get information on all current sessions.

Also note that the MaxRecordCount, ReturnTotalRecordCount, Skip, and SortBy parameters apply to GroupInfo records output rather than the filtered desktops.

**Related topics**

- Get-BrokerDesktop
- Group-Object

**Parameters**

- **-Uid**<Int32>
Gets desktops with a specific UID.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-Property <String>
Selects the property by which matching desktops are grouped.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-MachineName <String>
Gets desktops with a specific machine name (in the form 'domain\machine').

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AgentVersion <String>
Gets desktops with a specific Citrix Virtual Delivery Agent version.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-ApplicationInUse <String>
Gets desktops running a specified published application (identified by browser name).
String comparisons are case-insensitive.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AssignedClientName <String>
Gets desktops assigned to a specific client name.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AssignedIPAddress <String>
Gets desktops assigned to a specific IP address.
<table>
<thead>
<tr>
<th>-AssociatedUserFullName</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AssociatedUserFullName<String>**

Gets desktops with an associated user identified by their full name (usually in the form 'first-name last-name').

Associated users are the current user for shared desktops, and the assigned users for private desktops.

<table>
<thead>
<tr>
<th>-AssociatedUserName</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AssociatedUserName<String>**

Gets desktops with an associated user identified by their user name (in the form 'domain\user').

Associated users are the current user for shared desktops, and the assigned users for private desktops.

<table>
<thead>
<tr>
<th>-AssociatedUserUPN</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AssociatedUserUPN<String>**

Gets desktops with an associated user identified by their User Principle Name (in the form 'user@domain').

Associated users are the current user for shared desktops, and the assigned users for private desktops.

<table>
<thead>
<tr>
<th>-AutonomouslyBrokered</th>
<th>Boolean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AutonomouslyBrokered<Boolean>**

Gets desktops with the autonomously brokered session flag.

Session properties are always $null for multi-session desktops.

<table>
<thead>
<tr>
<th>-CatalogName</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-CatalogName<String>**

Gets desktops from the catalog with the specific name.

<table>
<thead>
<tr>
<th>-CatalogUid</th>
<th>Int32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-CatalogUid<Int32>**
Gets desktops from a catalog with a specific UID.

<table>
<thead>
<tr>
<th></th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ClientAddress**<string>

Gets desktops with a specific client IP address.

<table>
<thead>
<tr>
<th></th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ClientName**<string>

Gets desktops with a specific client name.

<table>
<thead>
<tr>
<th></th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ClientVersion**<string>

Gets desktops with a specific client version.

<table>
<thead>
<tr>
<th></th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ColorDepth**<ColorDepth>

Gets desktops configured with a specific color depth. Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

<table>
<thead>
<tr>
<th></th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ConnectedViaHostName**<string>

Gets desktops with a specific host name of the incoming connection. This is usually a proxy or Citrix Access Gateway server.

<table>
<thead>
<tr>
<th></th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ConnectedViaIP**<String>

Gets desktops with a specific IP address of the incoming connection.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ControllerDNSName&lt;String&gt;</td>
<td>Gets desktops with a specific DNS name of the controller they are registered with.</td>
</tr>
<tr>
<td>-DeliveryType&lt;DeliveryType&gt;</td>
<td>Gets desktops of a particular delivery type. Valid values are AppsOnly, DesktopsOnly, DesktopsAndApps</td>
</tr>
<tr>
<td>-Description&lt;String&gt;</td>
<td>Get desktops with a specific description.</td>
</tr>
</tbody>
</table>
| -DesktopCondition<String> | Gets desktop with an outstanding desktop condition condition. Valid values are:  
  - CPU: Indicates the machine has high CPU usage  
  - ICALatency: Indicates the network latency is high  
  - UPMLogonTime: Indicates that the profile load time was high |
| -DesktopGroupName<String> | Gets desktops from a desktop group with the specified name. |
### -DesktopGroupUid<Int32>
Gets desktops from a desktop group with the specified UID.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -DesktopKind<DesktopKind>
Deprecated: Use AllocationType parameter.
Gets desktops of a particular kind.
Valid values are Private, Shared.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -DeviceId<String>
Gets desktops with a specific client device ID.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -DNSName<String>
Get desktops with a specific DNS name.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -FunctionalLevel<FunctionalLevel>
Gets desktops with a specific FunctionalLevel.
Valid values are L5, L7

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -HardwareId<String>
Gets desktops with a specific client hardware ID.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |
### -HostedMachineId<String>

Gets desktops with a specific machine ID known to the hypervisor.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

### -HostedMachineName<String>

Gets desktops with a specific machine name known to the hypervisor.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

### -HostingServerName<String>

Gets desktops with a specific name of the hosting hypervisor server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

### -HypervisorConnectionName<String>

Gets desktops with a specific name of the hosting hypervisor connection.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

### -HypervisorConnectionUid<Int32>

Gets desktops with a specific UID of the hosting hypervisor connection.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

### -IconUid<Int32>

Gets desktops with a specific configured icon. Note that desktops with a null IconUid use the icon of the desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

### -ImageOutOfDate<Boolean>

Gets desktops if they have an ImageOutOfDate flag.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-InMaintenanceMode</td>
<td>Boolean</td>
<td>Gets desktops with a specific InMaintenanceMode setting.</td>
</tr>
<tr>
<td>-IPAddress</td>
<td>String</td>
<td>Gets desktops with a specific IP address.</td>
</tr>
<tr>
<td>-IsAssigned</td>
<td>Boolean</td>
<td>Gets desktops according to whether they are assigned or not.</td>
</tr>
<tr>
<td>-IsPhysical</td>
<td>Boolean</td>
<td>Specifies if machines in the catalog can be power managed by the Citrix Broker Service.</td>
</tr>
<tr>
<td>-LastConnectionFailure</td>
<td>ConnectionFailureReason</td>
<td>Gets desktops with a specific reason for the last recorded connection failure. Valid values are None, SessionPreparation, RegistrationTimeout, ConnectionTimeout, Licensing, Ticketing, and Other.</td>
</tr>
<tr>
<td>-LastConnectionTime</td>
<td>DateTime</td>
<td></td>
</tr>
</tbody>
</table>
Gets desktops that last connected at a specific time. This is the time that the broker detected that the connection attempt either succeeded or failed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**LastConnectionUser**<String>

Gets desktops where a specific user name last attempted a connection (in the form 'domain\user').

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**LastDeregistrationReason**<DeregistrationReason>

Gets desktops whose broker last recorded a specific deregistration reason.

Valid values are $null, AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**LastDeregistrationTime**<DateTime>

Gets desktops that were last deregistered by a specific time.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**LastErrorReason**<String>

Gets desktops with the specified last error reason.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**LastErrorTime**<DateTime>

Gets desktops with the specified last error time.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
### -LastHostingUpdateTime (DateTime)

Gets desktops with a specific time that the hosting information was last updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -LaunchedViaHostName (String)

Gets desktops with a specific host name of the StoreFront server from which the user launched the session.

Session properties are always $null for multi-session desktops.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -LaunchedViaIP (String)

Gets desktops with a specific IP address of the StoreFront server from which the user launched the session.

Session properties are always $null for multi-session desktops.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -MachineInternalState (MachineInternalState)

Gets desktops with the specified internal machine state.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -MachineUid (Int32)

Gets desktops with a specific machine UID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -OSType (String)

Gets desktops by the type of operating system they are running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -OSVersion (String)

Gets desktops with a specific version of the operating system.
**-PersistUserChanges:<PersistUserChanges>**

Gets desktops by the location where the user changes are persisted.
- OnLocal - User changes are persisted locally.
- Discard - User changes are discarded.
- OnPvd - User changes are persisted on the Pvd.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-PowerActionPending:<Boolean>**

Gets desktops with a specific power action pending state.
Valid values are $true or $false.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-PowerState:<PowerState>**

Gets desktops with a specific power state.
Valid values are Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Protocol:<String>**

Gets desktops with connections using a specific protocol, for example ‘HDX’, or ‘RDP’.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ProvisioningType:<ProvisioningType>**

Specifies the provisioning type for the catalog. Values can be:
- Manual - No provisioning.
- PVS - Machine provisioned by PVS (machine may be physical, blade, VM...).
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-MCS</td>
<td>string</td>
<td>MCS - Machine provisioned by MCS (machine must be VM).</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-PublishedApplication</td>
<td>string</td>
<td>Gets desktops with a specific application published on them (identified by its browser name).</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-PublishedName</td>
<td>string</td>
<td>Gets desktops with a specific published name.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-PvdStage</td>
<td>PvdStage</td>
<td>Gets machines with a specific personal vDisk stage.</td>
</tr>
<tr>
<td>Valid values</td>
<td>None, Requested, Starting and Working.</td>
<td>Valid values are None, Requested, Starting and Working.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-RegistrationState</td>
<td>RegistrationState</td>
<td>Gets desktops with a specific registration state.</td>
</tr>
<tr>
<td>Valid values</td>
<td>Unregistered, Initializing, Registered and AgentError.</td>
<td>Valid values are Unregistered, Initializing, Registered and AgentError.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-SecureIcaActive</td>
<td>boolean</td>
<td>Gets desktops depending on whether the current session uses SecureICA or not.</td>
</tr>
<tr>
<td>Session properties</td>
<td>$null</td>
<td>Session properties are always $null for multi-session desktops.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-SecureIcaRequired</td>
<td>boolean</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1079
Gets desktops configured with a particular SecureICARequired setting. Note that the desktop setting of $null indicates that the desktop group value is used.

Session properties are always $null for multi-session desktops.

| Required? | False |
| Default Value | |
| Accept Pipeline Input? | False |

-SessionHidden:Boolean

Gets desktops by whether their sessions are hidden or not. Hidden sessions are treated as though they do not exist when launching sessions; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

Session properties are always $null for multi-session desktops.

| Required? | False |
| Default Value | |
| Accept Pipeline Input? | False |

-SessionId: Int32

Deprecated.

Gets desktops by session ID, a unique identifier that Remote Desktop Services uses to track the session but it is only unique on that machine.

| Required? | False |
| Default Value | |
| Accept Pipeline Input? | False |

-SessionState:SessionState

Gets desktops with a specific session state.

Valid values are $null, Other, PreparingSession, Connected, Active, Disconnected, Reconnecting, NonBrokeredSession, and Unknown.

Session properties are always $null for multi-session desktops.

| Required? | False |
| Default Value | |
| Accept Pipeline Input? | False |

-SessionStateChangeTime:DateTime

Gets desktops whose sessions last changed state at a specific time.

Session properties are always $null for multi-session desktops.

| Required? | False |
| Default Value | |
| Accept Pipeline Input? | False |

-SessionUid: Int64

Gets desktops with a specific session UID ($null for no session).
Session properties are always $null for multi-session desktops.

<table>
<thead>
<tr>
<th>-SessionUserName &lt;String&gt;</th>
<th>Gets desktops with a specific user name for the current session (in the form 'domain\user').</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-SessionUserSID &lt;String&gt;</th>
<th>Gets desktops with a specific SID of the current session user.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-SID &lt;String&gt;</th>
<th>Gets desktops with a specific machine SID.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-SmartAccessTag &lt;String&gt;</th>
<th>Gets desktops where the session has the specific SmartAccess tag.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-StartTime &lt;DateTime&gt;</th>
<th>Gets desktops with a specific session start time.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
### -SummaryState
*DesktopSummaryState*

Gets desktops with a specific summary state.

Valid values are Off, Unregistered, Available, Disconnected, and InUse.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -Tag
*String*

Gets desktops with a specific tag.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -WillShutdownAfterUse
*Boolean*

Gets desktops depending on whether they shut down after use or not.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -ApplicationUid
*Int32*

Gets desktops with a specific published application (identified by its UID).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -ReturnTotalRecordCount
*SwitchParameter*

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -MaxRecordCount
*Int32*

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -Skip
*Int32*

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **SortBy**<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Filter**<String>

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdminSDK.GroupInfo

Each GroupInfo object represents one group, and contains the following properties:

-- Count: The count of desktops in this group.

-- Name: The value of the property the desktops were grouped by (as a string).

If you do not specify -SortBy, groups are sorted with the largest count first.

Notes

To compare dates or times, use -Filter and relative comparisons. For more information, see about_Broker_Filtering and the examples.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Group-BrokerDesktop -Property SummaryState -DesktopGroupName dg1
Group desktops from the dg1 group by summary state.
----------- EXAMPLE 2 --------------------------

C:\PS> Group-BrokerDesktop -Property LastConnectionFailure -Filter { LastConnectionFailure -ne "None" -and LastConnectionTime -ge '7' } -MaxRecordCount 1
For desktops where the last connection attempt failed, list the most common reason for failure, ignoring connections that failed over a week ago.

----------- EXAMPLE 3 --------------------------

C:\PS> Group-BrokerDesktop -Property HostingServerName -DesktopCondition ICALatency -SortBy Name
List alphabetically the hypervisor servers hosting desktops that are currently experiencing high network latency.
Group-BrokerMachine

Oct 10, 2013
Groups and counts machines with the same value for a specified property.

Syntax

Group-BrokerMachine [-Uid] <Int32> -Property <String> [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Filters machines using the specified criteria, then groups and counts matching machines with the same value for a particular property. The number of machines in the group, and the property value for the group, is output. For example:

C:\PS> Group-BrokerMachine -Property SummaryState

Count Name
----- ----
43 Available
17 InUse
3 Disconnected

Filtering supports the same options as the Get-BrokerMachine cmdlet, and allows filtering on both machine and session properties.

Group-BrokerMachine is similar to the standard PowerShell Group-Object, but is faster than piping the output of Get-BrokerMachine into Group-Object when working with many machines.

Note that the MaxRecordCount, ReturnTotalRecordCount, Skip, and SortBy parameters apply to GroupInfo records output rather than the filtered machines.

Related topics

Get-BrokerMachine

Group-Object

Parameters

-Uid <Int32>
## Gets a machine with a specific UID

| Required? | true |
| Default Value |  |
| Accept Pipeline Input? | false |

### -Property<String>

Selects the property by which matching machines are grouped.

| Required? | true |
| Default Value |  |
| Accept Pipeline Input? | false |

### -MachineName< String>

Gets machines with a specific machine name (in the form domain\machine).

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -AgentVersion< String>

Gets machines with a specific Virtual Delivery Agent version.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -AllocationType<AllocationType>

Gets machines from catalogs with the specified allocation type.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -ApplicationInUse< String>

Gets machines running a specified published application. String comparisons are case-insensitive.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -AssignedClientName< String>

 Gets machines that have been assigned to the specific client name.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |
### AssignedIPAddress

**Description:**
Gets machines that have been assigned to the specific client IP address.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### AssociatedUserFullName

**Description:**
Gets machines with an associated user identified by their full name (usually 'first-name last-name').
Associated users are all current users of a desktop, plus the assigned users for private desktops.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### AssociatedUserName

**Description:**
Gets machines with an associated user identified by their user name (in the form 'domain\user').
Associated users are all current users of a desktop, plus the assigned users for private desktops.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### AssociatedUserSID

**Description:**
Gets machines with an associated user identified by their Windows SID.
Associated users are all current users of a desktop, plus the assigned users for private desktops.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### AssociatedUserUPN

**Description:**
Gets machines with an associated user identified by their User Principle Name (in the form 'user@domain').
Associated users are all current users of a desktop, plus the assigned users for private desktops.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### CatalogName

**Description:**
Gets machines from the catalog with the specific name.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-CatalogUid&lt;Int32&gt;</td>
<td>Gets machines from the catalog with the specific UID.</td>
</tr>
<tr>
<td>-CatalogUUID&lt;Guid&gt;</td>
<td>Gets machines from the catalog with the specific UUID.</td>
</tr>
<tr>
<td>-ColorDepth&lt;ColorDepth&gt;</td>
<td>Gets machines configured with a specific color depth.</td>
</tr>
<tr>
<td>-ControllerDNSName&lt;String&gt;</td>
<td>Gets machines by the DNS name of the controller they are registered with.</td>
</tr>
<tr>
<td>-DeliveryType&lt;DeliveryType&gt;</td>
<td>Gets machines of a particular delivery type.</td>
</tr>
<tr>
<td>-Description&lt;String&gt;</td>
<td>Get machines by description.</td>
</tr>
</tbody>
</table>
**-DesktopCondition**<String>

Gets machines with an outstanding desktop condition.

Valid values are:
- CPU: Indicates the machine has high CPU usage
- ICALatency: Indicates the network latency is high
- UPMLogonTime: Indicates that the profile load time was high

**-DesktopGroupName**<String>

Gets machines from a desktop group with the specified name.

**-DesktopGroupUid**<Int32>

Gets machines from a desktop group with a specific UID.

**-DesktopGroupUUID**<Guid>

Gets machines from a desktop group with a specific UUID.

**-DesktopKind**<DesktopKind>

Deprecated: Use AllocationType parameter.

Gets machines of a particular kind.

Valid values are Private, Shared.
<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>DesktopUid</td>
<td>Int32</td>
<td>Gets the machine that corresponds to the desktop with the specific UID.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>DNSName</td>
<td>String</td>
<td>Gets machines with the specific DNS name.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>FaultState</td>
<td>MachineFaultState</td>
<td>Gets machines currently in the specified fault state.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>FunctionalLevel</td>
<td>FunctionalLevel</td>
<td>Gets machines with a specific FunctionalLevel.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td>Valid values are L5, L7</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>HostedMachineId</td>
<td>String</td>
<td>Gets machines with the specific machine ID known to the hypervisor.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>HostedMachineName</td>
<td>String</td>
<td>Gets machines with the specific machine name known to the hypervisor.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
-**HostingServerName**: `<String>`

 Gets machines by the name of the hosting hypervisor server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**HypervisorConnectionName**: `<String>`

 Gets machines with the specific name of the hypervisor connection hosting them.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**HypervisorConnectionUid**: `<Int32>`

 Gets machines with the specific UID of the hypervisor connection hosting them.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**HypHypervisorConnectionUid**: `<Guid>`

 Gets machines with the specific UUID of the hypervisor connection hosting them.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**IconUid**: `<Int32>`

 Gets machines by configured icon. Note that machines with a null IconUid use the icon of the desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**ImageOutOfDate**: `<Boolean>`

 Gets machines depending on whether their disk image is out of date or not (for machines provisioned using MCS only).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**InMaintenanceMode**: `<Boolean>`

 Gets machines by whether they are in maintenance mode or not.
<table>
<thead>
<tr>
<th>-IPAddress &lt;String&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

Gets machines with a specific IP address.

<table>
<thead>
<tr>
<th>-IsAssigned &lt;Boolean&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

Gets machines according to whether they are assigned or not. Machines may be assigned to one or more users or groups, a client IP address or a client endpoint name.

<table>
<thead>
<tr>
<th>-IsPhysical &lt;Boolean&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

Gets machines according to whether they can be power managed by XenDesktop or not.

<table>
<thead>
<tr>
<th>-LastConnectionFailure &lt;ConnectionFailureReason&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

Gets machines with a specific reason for the last recorded connection failure. This value is None if the last connection was successful or if there has been no attempt to connect to the machine yet.

Valid values are None, SessionPreparation, RegistrationTimeout, ConnectionTimeout, Licensing, Ticketing, and Other.

<table>
<thead>
<tr>
<th>-LastConnectionTime &lt;DateTime&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

Gets machines to which a user session connection occurred at a specific time. This is the time that the broker detected that the connection attempt either succeeded or failed.

<table>
<thead>
<tr>
<th>-LastConnectionUser &lt;String&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

Gets machines where a specific user name last attempted a connection (in the form 'domain\user').
- **LastDeregistrationReason**: `<DeregistrationReason>`

  Gets machines whose broker last recorded a specific deregistration reason.

  Valid values are $null, AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

- **LastDeregistrationTime**: `<DateTime>`

  Gets machines by the time that they were last deregistered.

- **LastErrorReason**: `<String>`

  Gets machines with the specified last error reason.

- **LastErrorTime**: `<DateTime>`

  Gets machines with the specified last error time.

- **LastHostingUpdateTime**: `<DateTime>`

  Gets machines with a specific time that the hosting information was last updated.

- **LoadIndex**: `<Int32>`
Gets machines by their current load index.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-MachineInternalState

Gets machines with the specified internal state.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: `-Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-OSType

Gets machines by the type of operating system they are running.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-OSVersion

Gets machines by the version of the operating system they are running.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-PersistUserChanges

Gets machines according to the location where user changes are persisted. Values can be:

- OnLocal - User changes are persisted locally.
- Discard - User changes are discarded.
- OnPvd - User changes are persisted on the Pvd.
- **Accept Pipeline Input?**

  Gets machines depending on whether a power action is pending or not.

  Valid values are $true or $false.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **PowerActionPending**<Boolean>

  Gets machines with a specific power state.

  Valid values are Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **PowerState**<PowerState>

  Specifies the provisioning type for the catalog. Values can be:

  o Manual - No provisioning.
  o PVS - Machine provisioned by PVS (machine may be physical, blade, VM,...).
  o MCS - Machine provisioned by MCS (machine must be VM).

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **ProvisioningType**<ProvisioningType>

  Gets desktops with a specific published name.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **PublishedName**<String>

  Gets machines at a specific personal vDisk stage.
Valid values are None, Requested, PoweringOn and Working.

<table>
<thead>
<tr>
<th>-RegistrationState&lt;RegistrationState&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gets machines in a specific registration state.</td>
</tr>
<tr>
<td>Valid values are Unregistered, Initializing, Registered, and AgentError.</td>
</tr>
</tbody>
</table>

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-ScheduledReboot <ScheduledReboot>

Gets machines according to their current status with respect to any scheduled reboots (for either scheduled desktop group reboots or image rollout purposes). Valid values are:

- None - No reboot currently scheduled.
- Pending - Reboot scheduled but machine still available for use.
- Draining - Reboot scheduled. New logons are disabled, but reconnections to existing sessions are allowed.
- InProgress - Machine is actively being rebooted.
- Natural - Natural reboot in progress. Machine is awaiting a restart.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-SecureIcaRequired <Boolean>

Gets machines configured with a particular SecureIcaRequired setting. Note that the machine setting of $null indicates that the desktop group value is used.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-SessionAutonomouslyBrokered <Boolean>

Gets machines with the autonomously brokered session flag.

Session properties are always $null for multi-session machines.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-SessionClientAddress <String>

Gets machines with a specific client IP address.
### -SessionClientName

**Gets machines with a specific client name.**

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -SessionClientVersion

**Gets machines with a specific client version.**

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -SessionConnectedViaHostName

**Gets machines with a specific host name of the incoming connection. This is usually a proxy or Citrix Access Gateway server.**

Session properties are always null for multi-session machines.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -SessionConnectedViaIP

**Gets machines with a specific IP address of the incoming connection.**

Session properties are always null for multi-session machines.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -SessionCount

**Gets machines according to the total number of both pending and established user sessions on the machine.**

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -SessionDeviceId

**Gets machines with a specific client device ID.**
### -SessionHardwareId<String>

Gets machines with a specific client hardware ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>False</td>
</tr>
</tbody>
</table>

### -SessionHidden<Boolean>

Gets machines by whether their sessions are hidden or not. Hidden sessions are treated as though they do not exist when launching sessions using XenDesktop; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

<table>
<thead>
<tr>
<th>Required?</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>False</td>
</tr>
</tbody>
</table>

### -SessionKey<Guid>

Gets machine running the session with the specified unique key.

Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>False</td>
</tr>
</tbody>
</table>

### -SessionLaunchedViaHostName<String>

Gets machines with a specific host name of the Web Interface server from which the user launched the session.

Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>False</td>
</tr>
</tbody>
</table>

### -SessionLaunchedViaIP<String>

Gets machines with a specific IP address of the Web Interface server from which the user launched the session.

Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>False</td>
</tr>
</tbody>
</table>

### -SessionProtocol<String>


Gets machines with connections using a specific protocol, for example 'HDX', or 'RDP'.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-SessionSecureIcaActive</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-SessionSecureIcaActive<Boolean>

Gets machines depending on whether the current session uses SecureICA or not. Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-SessionsEstablished</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-SessionsEstablished<Int32>

Gets machines according to the number of established user sessions present on the machine.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-SessionSmartAccessTag</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-SessionSmartAccessTag<String>

Gets session machines where the session has the specific SmartAccess tag. Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-SessionsPending</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-SessionsPending<Int32>

Get machines according to the number of pending user sessions for the machine.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-SessionStartTime</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-SessionStartTime<DateTime>

Gets machines with a specific session start time. Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-SessionState</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-SessionState<SessionState>
**-SessionState**

Gets machines with a specific session state.

Valid values are $null, Other, PreparingSession, Connected, Active, Disconnected, Reconnecting, NonBrokeredSession, and Unknown.

Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionStateChangeTime**<DateTime>

Gets machines whose sessions last changed state at a specific time.

Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionSupport**<SessionSupport>

Gets machines that have the specified session capability. Values can be:

- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionUid**<Int64>

Gets single-session machines with a specific session UID ($null for no session).

Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionUserName**<String>

Gets machines with a specific user name for the current session (in the form 'domain\user').

Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SessionUserSID**<String>

Gets machines with a specific SID of the current session user.
Session properties are always $null for multi-session machines.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-SID&lt;String&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>Gets machines with a specific machine SID.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-SummaryState&lt;DesktopSummaryState&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>Gets machines with a specific summary state.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Valid values are Off, Unregistered, Available, Disconnected, and InUse.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-SupportedPowerActions&lt;String[]&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>A list of power actions supported by this machine.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Tag&lt;String&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>Gets machines where the session has the given SmartAccess tag.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-UUID&lt;Guid&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>Gets machines with the specified value of UUID.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-VMToolsState&lt;VMToolsState&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>Gets machines with a specific VM tools state.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Valid values are NotPresent, Unknown, NotStarted, and Running.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>WillShutdownAfterUse</strong>&lt;Boolean&gt;</td>
<td>Gets machines depending on whether they shut down after use or not.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>WindowsConnectionSetting</strong>&lt;WindowsConnectionSetting&gt;</td>
<td>Gets machines according to their current Windows connection setting (logon mode). Valid values are:</td>
</tr>
<tr>
<td>o LogonEnabled - All logons are enabled.</td>
<td></td>
</tr>
<tr>
<td>o Draining - New logons are disabled, but reconnections to existing sessions are allowed.</td>
<td></td>
</tr>
<tr>
<td>o DrainingUntilRestart - Same as Draining, but setting reverts to LogonEnabled when machine next restarts.</td>
<td></td>
</tr>
<tr>
<td>o LogonDisabled - All logons and reconnections are disabled.</td>
<td></td>
</tr>
<tr>
<td>This is a Windows setting and is not controlled by XenDesktop. It applies only to multi-session machines; for single-session machines its value is always LogonEnabled.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AssignedUserSID</strong>&lt;String&gt;</td>
<td>Gets machines with the specific SID of the user to whom the desktop is assigned.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ReturnTotalRecordCount</strong>&lt;SwitchParameter&gt;</td>
<td>When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MaxRecordCount</strong>&lt;Int32&gt;</td>
<td>Specifies the maximum number of records to return.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
</tbody>
</table>
Accept Pipeline Input?  

**-Skip<Int32>**

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |

Accept Pipeline Input?  

**-SortBy<String>**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |

Accept Pipeline Input?  

**-Filter<String>**

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

| Required? | false |

Accept Pipeline Input?  

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |

Accept Pipeline Input?  

**Input Type**

None You cannot pipe input into this cmdlet.

**Return Values**

Citrix.BrokerAdminSDK.GroupInfo

Each GroupInfo object represents one group, and contains the following properties:

-- Count: The count of machines in this group.

-- Name: The value of the property the machines were grouped by (as a string).

If you do not specify -SortBy, groups are sorted with the largest count first.

**Notes**

To compare dates or times, use -Filter and relative comparisons. For more information, see about_Broker_Filtering and the examples.

**Examples**

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Group-BrokerMachine -Property SummaryState -DesktopGroupName dg1
Group machines from the dg1 group by summary state.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Group-BrokerMachine -Property LastConnectionFailure -Filter { LastConnectionFailure -ne "None" -and LastConnectionTime -ge '-7' } -MaxRecordCount 1
For machines where the last connection attempt failed, list the most common reason for failure, ignoring connections that failed over a week ago.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Group-BrokerMachine -Property HostingServerName -DesktopCondition ICALatency -SortBy Name
List alphabetically the hypervisor servers hosting machines that are currently experiencing high network latency.
Import-BrokerDesktopPolicy

Oct 10, 2013
Sets the site wide Citrix Group Policy settings for the site.

Syntax

Import-BrokerDesktopPolicy [-Policy] <Byte[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Import-BrokerDesktopPolicy sets the site wide Citrix Group Policy settings. A successful call to this cmdlet will result in the supplied data being uploaded to every machine in the site prior to its next session launch.

Related topics

Export-BrokerDesktopPolicy
New-BrokerConfigurationSlot
New-BrokerMachineConfiguration

Parameters

- **Policy**: <Byte[]>
  The configuration data containing the Citrix Group Policy settings to apply to every machine in the site.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **LoggingId**: <Guid>
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**: <String>
  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.
Required? | false
---|---
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
Accept Pipeline Input? | false

**Input Type**

System.Byte[] The configuration data as an opaque binary blob.

**Return Values**

None

**Notes**

Import-BrokerDesktopPolicy performs a specialized operation. Direct usage of it in scripts is discouraged, and could result in data corruption. It is recommended that this operation be performed via the Citrix Studio.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Import-BrokerDesktopPolicy $policyData
This command sets the Citrix Group Policy settings in the site. These policy settings are then applied to every machine prior to the next session launch.
New-BrokerAccessPolicyRule

Oct 10, 2013
Creates a new rule in the site's access policy.

Syntax


Detailed Description

The New-BrokerAccessPolicyRule cmdlet adds a new rule to the site's access policy.

An access policy rule defines a set of connection filters and access control rights relating to a desktop group. These allow fine-grained control of what access is granted to a desktop group based on details of, for example, a user's endpoint device, its address, and the user's identity.

Multiple rules in the access policy can apply to the same desktop group.

For a user to gain access to a desktop group via a rule their connection must match all its enabled include filters, and none of its enabled exclude filters. In addition, for a user to be able to launch a desktop or application resource session from the desktop group, they must have an entitlement to use the resource granted by the entitlement or assignment policies, or by direct machine assignment.

Related topics

Get-BrokerAccessPolicyRule
Set-BrokerAccessPolicyRule
Rename-BrokerAccessPolicyRule
Remove-BrokerAccessPolicyRule

Parameters

-Name <String>

Specifies the administrative name of the new rule. Each rule within the site's access policy must have a unique name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

Accept Pipeline Input? true (ByPropertyName)

-DesktopGroupUid <Int32>

Specifies the desktop group to which the new rule applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

Accept Pipeline Input? true (ByPropertyName)

-IncludedDesktopGroups <DesktopGroup[]>

This parameter is supported for backward compatibility only. If used only a single desktop group UID can be specified.

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1107
The IncludedDesktopGroups and IncludedDesktopGroupFilterEnabled parameters have been superseded by the DesktopGroupUid parameter.

| **Required?** | **true** |
| **Default Value** | *(empty list)* |
| **Accept Pipeline Input?** | *(ByPropertyName)* |

**-AllowedConnections<AllowedConnection>**

Specifies whether connections must be local or via Access Gateway, and if so whether specified SmartAccess tags must be provided by Access Gateway with the connection. This property forms part of the included SmartAccess tags filter.

Valid values are Filtered, NotViaAG, and ViaAG.

For a detailed description of this property see "help about_Broker_AccessPolicy".

| **Required?** | **false** |
| **Default Value** | *(empty list)* |
| **Accept Pipeline Input?** | *(ByPropertyName)* |

**-AllowedProtocols<String[]>**

Specifies the protocols (for example HDX, RDP) available to the user for sessions delivered from the new rule's desktop group. If the user gains access to a desktop group by multiple rules, the allowed protocol list is the combination of the protocol lists from all those rules.

If the protocol list is empty, access to the desktop group is implicitly denied.

| **Required?** | **false** |
| **Default Value** | HDX |
| **Accept Pipeline Input?** | *(ByPropertyName)* |

**-AllowedUsers<AllowedUser>**

Specifies the behavior of the included users filter of the new rule. This can restrict access to a list of named users or groups, or allow access to any authenticated user. For a detailed description of this property see "help about_Broker_AccessPolicy".

Valid values are Filtered, AnyAuthenticated, and Any.

| **Required?** | **false** |
| **Default Value** | *(empty list)* |
| **Accept Pipeline Input?** | *(ByPropertyName)* |

**-AllowRestart<Boolean>**

Specifies if the user can restart sessions delivered from the new rule's desktop group. Session restart is handled as follows: For sessions on single-session power-managed machines, the machine is powered off, and a new session launch request made; for sessions on multi-session machines, a logoff request is issued to the session, and a new session launch request made; otherwise the property is ignored.

| **Required?** | **false** |
| **Default Value** | *(empty list)* |
| **Accept Pipeline Input?** | *(ByPropertyName)* |

**-Description<String>**

Specifies an optional description of the new rule. The text is purely informational for the administrator, it is never visible to the end user.

| **Required?** | **false** |
| **Default Value** | *(empty list)* |
| **Accept Pipeline Input?** | *(ByPropertyName)* |
### -Enabled<Optional<Boolean>>

Specifies whether the new rule is initially enabled. A disabled rule is ignored when evaluating the site's access policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>true</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -ExcludedClientIPFilterEnabled<Optional<Boolean>>

Specifies whether the excluded client IP address filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -ExcludedIPs<Optional<String[]>>

Specifies IP addresses of user devices explicitly denied access to the new rule's desktop group. Addresses can be specified as simple numeric addresses or as subnet masks (for example, 10.40.37.5 or 10.40.0.0/16). This property forms part of the excluded client IP address filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>(empty list)</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -ExcludedClientNameFilterEnabled<Optional<Boolean>>

Specifies whether the excluded client names filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -ExcludedClientNames<Optional<String[]>>

Specifies names of user devices explicitly denied access to the new rule's desktop group. This property forms part of the excluded client names filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>(empty list)</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -ExcludedSmartAccessFilterEnabled<Optional<Boolean>>

Specifies whether the excluded SmartAccess tags filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -ExcludedSmartAccessTags<Optional<String[]>>

Specifies SmartAccess tags which explicitly deny access to the new rule's desktop group if any occur in those provided by Access Gateway with the user's connection. This property forms part of the excluded SmartAccess tags filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>(empty list)</td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | true (ByPropertyName)
---|---

### -ExcludedUserFilterEnabled<Boolean>

Specifies whether the excluded users filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

| Required? | false |
| Default Value | false |
| Accept Pipeline Input? | true (ByPropertyName) |

### -ExcludedUsers<user[]>

Specifies any users and groups who are explicitly denied access to the new rule's desktop group. This property forms part of the excluded users filter.

| Required? | false |
| Default Value | (empty list) |
| Accept Pipeline Input? | true (ByPropertyName) |

### -HdxSslEnabled<Boolean>

Indicates whether SSL encryption is enabled for sessions delivered from the rule's desktop group.

| Required? | false |
| Default Value | $false |
| Accept Pipeline Input? | true (ByPropertyName) |

### -IncludedClientIPFilterEnabled<Boolean>

Specifies whether the included client IP address filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

| Required? | false |
| Default Value | false |
| Accept Pipeline Input? | true (ByPropertyName) |

### -IncludedClientIPs<IPAddressRange>[

Specifies IP addresses of user devices allowed access to the new rule's desktop group. Addresses can be specified as simple numeric addresses or as subnet masks (for example, 10.40.37.5 or 10.40.0.0/16). This property forms part of the included client IP address filter.

| Required? | false |
| Default Value | (empty list) |
| Accept Pipeline Input? | true (ByPropertyName) |

### -IncludedClientNameFilterEnabled<Boolean>

Specifies whether the included client name filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

| Required? | false |
| Default Value | false |
| Accept Pipeline Input? | true (ByPropertyName) |

### -IncludedClientNames<String>[

Specifies names of user devices allowed access to the new rule's desktop group. This property forms part of the included client names filter.

| Required? | false |
| Default Value | (empty list) |
Accept Pipeline Input? | true (ByPropertyName)
---|---

INCLUDEDSMARTACCESSFILTERENABLED | Boolean
Specifies whether the included SmartAccess tags filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

| Required? | false |
| Default Value | false |
| Accept Pipeline Input? | true (ByPropertyName) |

INCLUDEDSMARTACCESSTAGS | String[]
Specifies SmartAccess tags which grant access to the new rule's desktop group if any occur in those provided by Access Gateway with the user's connection. This property forms part of the excluded SmartAccess tags filter.

| Required? | false |
| Default Value | (empty list) |
| Accept Pipeline Input? | true (ByPropertyName) |

INCLUDEDUSERFILTERENABLED | Boolean
Specifies whether the included users filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

| Required? | false |
| Default Value | false |
| Accept Pipeline Input? | true (ByPropertyName) |

INCLUDEDUSERS | User[]
Specifies users and groups who are granted access to the new rule's desktop group. This property forms part of the included users filter.

| Required? | false |
| Default Value | (empty list) |
| Accept Pipeline Input? | true (ByPropertyName) |

loggingId | Guid
Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

ADMINADDRESS | String
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

INCLUDEDDESKTOPGROUPFILTERENABLED | Boolean
This parameter is supported for backward compatibility only. If used the supplied value must be $true.

The INCLUDEDDESKTOPGROUPS and INCLUDEDDESKTOPGROUPFILTERENABLED parameters have been superseded by the DesktopGroupUid parameter.
Required? | false
---|---
Default Value | true
Accept Pipeline Input? | true (ByPropertyName)

Input Type
None You cannot pipe input into this cmdlet.

Return Values
Citrix.BrokerAdmin.SDK.AccessPolicyRule
New-BrokerAccessPolicyRule returns the newly created access policy rule.

Examples
-------------------------- EXAMPLE 1 --------------------------
C:\PS> $dg = Get-BrokerDesktopGroup 'Tech Support'
C:\PS> New-BrokerAccessPolicyRule 'UK Tech Support' -IncludedUserFilterEnabled $true -IncludedUsers support\uk-staff -DesktopGroupUid $dg.Uid -AllowedProtocols 'HDX'

Creates an access policy rule allowing access to the Tech Support desktop group for all users of the SUPPORT\uk-staff group. Connections to desktop or application resources in the group can only be made using the HDX protocol.

For users to gain access to resources in the group also requires that, depending on the desktop kind of the group, appropriate assignment or entitlement policy rules, or explicit machine assignments exist.
New-BrokerAppAssignmentPolicyRule

Oct 10, 2013
Creates a new application rule in the site's assignment policy.

Syntax


Detailed Description

The New-BrokerAppAssignmentPolicyRule cmdlet adds a new application rule to the site's assignment policy.

An application rule in the assignment policy defines the users who are entitled to a self-service persistent machine assignment from the rule's desktop group; once assigned the machine can run one or more applications published from the group.

The following constraints apply when creating an application assignment rule for a desktop group:

- The group's desktop kind must be Private
- The group's delivery type must be AppsOnly
- Only a single application rule can apply to a given group
- Application assignment rules cannot be applied to RemotePC groups.

When a user selects an application published from a private group, a currently unassigned machine is selected from the group and permanently assigned to the user. An application session is then launched to the machine. Subsequent launches are routed directly to the now assigned machine.

Once a machine has been assigned in this way, the original assignment rule plays no further part in access to the machine.

Related topics

Get-BrokerAppAssignmentPolicyRule
Set-BrokerAppAssignmentPolicyRule
Rename-BrokerAppAssignmentPolicyRule
Remove-BrokerAppAssignmentPolicyRule

Parameters

-Name <String>

Specifies the administrative name of the new application rule. Each rule in the site's assignment policy must have a unique name (irrespective of whether they are desktop or application rules).

<p>| Required? | true |</p>
<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-DesktopGroupUid**<int32>

Specifies the unique ID of the desktop group to which the new application rule applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

**-Description**<String>

Specifies an optional description of the new application rule. The text is purely informational for the administrator, it is never visible to the end user.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>null</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Enabled**<Boolean>

Specifies whether the new application rule is initially enabled. A disabled rule is ignored when evaluating the site's assignment policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>true</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-ExcludedUserFilterEnabled**<Boolean>

Specifies whether the excluded users filter is initially enabled. If the filter is disabled then any user entries in the filter are ignored when assignment policy rules are evaluated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-ExcludedUsers**<User[]>
Specifies the excluded users filter of the new application rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from the rule.

This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>(empty list)</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-IncludedUserFilterEnabled**<Boolean>

Specifies whether the included users filter is initially enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a machine assignment by the new application rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>true</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-IncludedUsers**<User[]>

Specifies the included users filter of the new application rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>(empty list)</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.AppAssignmentPolicyRule

New-BrokerAppAssignmentPolicyRule returns the newly created application rule in the assignment policy.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $dg = Get-BrokerDesktopGroup 'Sales Support'
C:\PS> New-BrokerAppAssignmentPolicyRule 'UK Office' -DesktopGroupUid $dg.Uid -IncludedUsers sales\uk-staff

Creates an application rule in the assignment policy that grants all members of the SALES\uk-staff group an entitlement to a single machine from the Sales Support desktop group. The machine can be used for running applications published from the group.
New-BrokerAppEntitlementPolicyRule

Oct 10, 2013
Creates a new application rule in the site's entitlement policy.

Syntax


Detailed Description

The New-BrokerAppEntitlementPolicyRule cmdlet adds a new application rule to the site's entitlement policy.

An application rule in the entitlement policy defines the users who are allowed per-session access to a machine to run one or more applications published from the rule's desktop group.

The following constraints apply when creating an application entitlement rule for a desktop group:

- The group's desktop kind must be Shared
- The group's delivery type must be AppsOnly or DesktopsAndApps
- Only a single application rule can apply to a given group

When a user selects an application published from a shared group, a machine is selected from the group on which to run the application. No permanent association exists between the user and the selected machine; once the session ends the association also ends.

Even though only a single application entitlement and therefore session can be defined for a group, the user can still run multiple applications from the group because the applications run within the same session.

Related topics

Get-BrokerAppEntitlementPolicyRule
Set-BrokerAppEntitlementPolicyRule
Rename-BrokerAppEntitlementPolicyRule
Remove-BrokerAppEntitlementPolicyRule

Parameters

-Name <String>

Specifies the administrative name of the new application rule. Each rule in the site's entitlement policy must have a unique name (irrespective of whether they are desktop or application rules).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1117
Accept Pipeline Input? | true (ByPropertyName)

-**DesktopGroupUid**<Int32>

Specifies the unique ID of the desktop group to which the new application rule applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**Description**<String>

Specifies an optional description of the new application rule. The text is purely informational for the administrator, it is never visible to the end user.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**Enabled**<Boolean>

Specifies whether the new application rule is initially enabled. A disabled rule is ignored when evaluating the site's entitlement policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>true</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**ExcludedUserFilterEnabled**<Boolean>

Specifies whether the excluded users filter is initially enabled. If the filter is disabled then any user entries in the filter are ignored when entitlement policy rules are evaluated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**ExcludedUsers**<User[]>

Specifies the excluded users filter of the application rule, that is, the users and groups who are explicitly denied entitlements to published applications from the desktop group.
This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>(empty list)</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
<td></td>
</tr>
</tbody>
</table>

**-IncludedUserFilterEnabled**<Boolean>

Specifies whether the included users filter is initially enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to an application session by the new rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
<td></td>
</tr>
</tbody>
</table>

**-IncludedUsers**<User[]>

Specifies the included users filter of the application rule, that is, the users and groups who are granted an entitlement to an application session by the new rule.

If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>(empty list)</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
<td></td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>


Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.AppEntitlementPolicyRule

New-BrokerAppEntitlementPolicyRule returns the newly created application rule in the entitlement policy.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerDesktopGroup 'Customer Support'
C:\PS> New-BrokerAppEntitlementPolicyRule 'UK Office' -DesktopGroupUid $dg.Uid -IncludedUsers support\uk-staff

Creates an application rule in the entitlement policy that entitles all members of the SUPPORT\uk-staff group to a machine for running applications published from the Customer Support desktop group.
New-BrokerApplication

Oct 10, 2013
Creates a new published application.

Syntax


Detailed Description

The New-BrokerApplication cmdlet creates a new published application in the site.

New-BrokerApplication creates the application object, and associates it with a desktop group. Application objects have three names that identify them (in addition to their Uid): the Name, BrowserName and the PublishedName. The BrowserName is unique across the entire site, and is primarily used internally. The Name is also unique and is what is seen by the administrator. The PublishedName is not unique and is what is seen by the users.

You can create both HostedOnDesktop and InstalledOnClient applications but the ApplicationType cannot be changed later.

The following special characters are not allowed in either the Name, BrowserName or the PublishedName: / : # * ? = < > | 

See about_Broker_Applications for more information.

Related topics

Add-BrokerApplication
Remove-BrokerApplication
Get-BrokerApplication
Remove-BrokerApplication
Rename-BrokerApplication
Set-BrokerApplication

Parameters

-Name <String>

Specifies the unique name of the application.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-CommandLineExecutable <String>

Specifies the name of the executable file to launch. The full path need not be provided if it's already in the path. Environment variables can also be used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1121
-DesktopGroup=DesktopGroup>

Specifies which desktop group this application should be associated with. The association between application and desktop groups can be added or removed using the Add-BrokerApplication and Remove-BrokerApplication cmdlets.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-ApplicationType=ApplicationType>

Specifies the type of the application: HostedOnDesktop or InstalledOnClient.

| Required? | false |
| Default Value | (required) |
| Accept Pipeline Input? | true (ByPropertyName) |

-BrowserName=String>

Specifies the internal name for this application. It must be unique in the site.

| Required? | false |
| Default Value | (same as Name) |
| Accept Pipeline Input? | true (ByPropertyName) |

-ClientFolder=String>

Specifies the folder that the application belongs to as the user sees it. This is the application folder that is seen in the Citrix Online Plug-in, in Web Services, and also in the end-user's Start menu. Subdirectories can be specified with '\ ' character. The following special characters are not allowed: / * ? < > | " :.

Note that this property cannot be set for applications of type InstalledOnClient.

| Required? | false |
| Default Value | null |
| Accept Pipeline Input? | true (ByPropertyName) |

-CommandLineArguments=String>

Specifies the command-line arguments to use when launching the executable. Environment variables can be used.

| Required? | false |
| Default Value | null |
| Accept Pipeline Input? | true (ByPropertyName) |

-CpuPriorityLevel=CpuPriorityLevel>

Specifies the CPU priority for the launched process. Valid values are: Low, BelowNormal, Normal, AboveNormal, and High. Note that this property cannot be set for applications of type InstalledOnClient.

| Required? | false |
| Default Value | Normal |
- **Description**: `String`<br>Specifies the description of the application. This is only seen by Citrix administrators and is not visible to users.<br>Required? false<br>Default Value null<br>Accept Pipeline Input? true (ByPropertyName)<br>

- **Enabled**: `Boolean`<br>Specifies whether or not this application can be launched.<br>Required? false<br>Default Value true<br>Accept Pipeline Input? true (ByPropertyName)<br>

- **IconFromClient**: `Boolean`<br>Specifies if the app icon should be retrieved from the application on the client. This is reserved for possible future use, and all applications of type HostedOnDesktop cannot set or change this value.<br>Required? false<br>Default Value false<br>Accept Pipeline Input? true (ByPropertyName)<br>

- **IconUid**: `Int32`<br>Specifies which icon to use for this application. This icon is visible both to the administrator (in the consoles) and to the user. If no icon is specified, then a generic built-in application icon is used.<br>Required? false<br>Default Value 2<br>Accept Pipeline Input? true (ByPropertyName)<br>

- **Priority**: `Int32`<br>Specifies the priority of the mapping between the application and desktop group. A value of zero has the highest priority, with increasing values indicating lower priorities.<br>Required? false<br>Default Value<br>Accept Pipeline Input? true (ByPropertyName)<br>

- **PublishedName**: `String`<br>The name seen by end users who have access to this application.<br>Required? false<br>Default Value The same value as that supplied for the name of the application.
**Accept Pipeline Input?**  
true (ByPropertyName)

**-SecureCmdLineArgumentsEnabled<Boolean>**

Specifies whether the command-line arguments are secured or not. This is reserved for possible future use, and all applications of type HostedOnDesktop can only have this value set to true.

| Required? | false |
| Default Value | true |
| Accept Pipeline Input? | true (ByPropertyName) |

**-ShortcutAddedToDesktop<Boolean>**

Specifies whether or not a shortcut to the application should be placed on the user device. This is valid only for the Citrix Online Plug-in.

| Required? | false |
| Default Value | false |
| Accept Pipeline Input? | true (ByPropertyName) |

**-ShortcutAddedToStartMenu<Boolean>**

Specifies whether a shortcut to the application should be placed in the user's start menu on their user device.

| Required? | false |
| Default Value | false |
| Accept Pipeline Input? | true (ByPropertyName) |

**-StartMenuFolder<String>**

Specifies the name of the start menu folder that holds the application shortcut (if any). This is valid only for the Citrix Online Plug-in. Subdirectories can be specified with `\` character. The following special characters are not allowed: `/ * ? < > | " :.

| Required? | false |
| Default Value | null |
| Accept Pipeline Input? | true (ByPropertyName) |

**-UserFilterEnabled<Boolean>**

Specifies whether the application's user filter is enabled or disabled. Where the user filter is enabled, the application is visible only to users who appear in the filter (either explicitly or by virtue of group membership).

| Required? | false |
| Default Value | false |
| Accept Pipeline Input? | true (ByPropertyName) |

**-UUID<Guid>**

An optional GUID for this application.

| Required? | false |
### -Visible<boolean>

Specifies whether or not this application is visible to users. Note that it's possible for an application to be disabled and still visible.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>true</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -WaitForPrinterCreation<boolean>

Specifies whether or not the session waits for the printers to be created before allowing the user to interact with the session. Note that this property cannot be set for applications of type InstalledOnClient.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -WorkingDirectory<string>

Specifies which working directory the executable is launched from. Environment variables can be used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -LoggingId<guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress<string>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Depends on parameter. Parameters can be piped by property name.

**Return Values**
C:PS> New-BrokerApplication -ApplicationType HostedOnDesktop -Name "Notepad" -CommandLineExecutable "notepad.exe" -DesktopGroup PrivateDG1
Creates and returns an object for a published application called "Notepad" that launches "notepad.exe".

-------------------------- EXAMPLE 2 --------------------------
C:PS> $dg = Get-BrokerDesktopGroup "SharedDG1"
C:PS> $app = New-BrokerApplication -ApplicationType HostedOnDesktop -Name "Notepad" -CommandLineExecutable "notepad.exe" -DesktopGroup $dg
C:PS> $group = Get-BrokerDesktopGroup -Name "Shared desktop group"
C:PS> Add-BrokerApplication $app -DesktopGroup $group
C:PS> $fta = Get-BrokerImportedFTA -ExtensionName ".txt"
C:PS> New-BrokerConfiguredFTA -ImportedFTA $fta -ApplicationUid $app.Uid
This is a much more complete example. It creates an application object to publish Notepad and associates it first with the "SharedDG1" desktop group. Next it adds an additional desktop group (one that can host applications), and publishes the application to that desktop group. It then gets the ImportedFTA object for the .txt file-type extension (this assumes file-type associations have already been imported), and then configures it so that ".txt" is associated with the published application.

Note: The appropriate access policy and app assignment/entitlement rules must also be configured to allow access to the application.
New-BrokerAssignmentPolicyRule

Oct 10, 2013
Creates a new desktop rule in the site's assignment policy.

Syntax


Detailed Description

The New-BrokerAssignmentPolicyRule cmdlet adds a new desktop rule to the site's assignment policy.

A desktop rule in the assignment policy defines the users who are entitled to self-service persistent machine assignments from the rule's desktop group. A rule defines how many machines a user is allowed from the group for delivery of full desktop sessions.

The following constraints apply when creating a desktop assignment rule for a desktop group:

- The group's desktop kind must be Private
- The group's delivery type must be DesktopsOnly
- Only one desktop assignment rule can be created for RemotePC groups.

When a user selects a machine assignment entitlement from a private group, a currently unassigned machine is selected from the group and permanently assigned to the user to create an assigned desktop. A desktop session is then launched to the machine. Subsequent launches are routed directly to the now assigned machine.

Once a machine has been assigned in this way, the original assignment rule plays no further part in access to the new desktop.

Multiple desktop rules in the assignment policy can apply to the same desktop group. Where a user is granted entitlements by more than one rule for the same group, they can have as many machine assignments from the group as the total of their entitlements.

Related topics

Get-BrokerAssignmentPolicyRule
Set-BrokerAssignmentPolicyRule
Rename-BrokerAssignmentPolicyRule
Remove-BrokerAssignmentPolicyRule

Parameters

-Name <String>

Specifies the administrative name of the new desktop rule. Each rule in the site's assignment policy must have a unique name (irrespective of whether they are desktop or application rules).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-DesktopGroupUid <Int32>
Specifies the unique ID of the desktop group to which the new desktop rule applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-ColorDepth**<ColorDepth>

Specifies the color depth of any desktop sessions to machines assigned by the new rule.

Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null (dynamically inherited from the desktop group)</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Description**<String>

Specifies an optional description of the new desktop rule. The text may be visible to the end user, for example, as a tooltip associated with the desktop entitlement.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null (dynamically inherited from the desktop group)</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Enabled**<Boolean>

Specifies whether the new desktop rule is initially enabled. A disabled rule is ignored when evaluating the site’s assignment policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>true</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-ExcludedUserFilterEnabled**<Boolean>

Specifies whether the excluded users filter is initially enabled. If the filter is disabled then any user entries in the filter are ignored when assignment policy rules are evaluated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-ExcludedUsers**<User[]>
Specifies the excluded users filter of the new desktop rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from the rule.

This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>(empty list)</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-IconUid<Int32>**

Specifies the unique ID of the icon used to display the machine assignment entitlement to the user, and of the assigned desktop itself following the assignment.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>null (dynamically inherited from the desktop group)</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-IncludedUserFilterEnabled<Boolean>**

Specifies whether the included users filter is initially enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a machine assignment by the new desktop rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

For rules that relate to RemotePC desktop groups however, if the included user filter is disabled, the rule is effectively disabled.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>true</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-IncludedUsers<User[]>**

Specifies the included users filter of the new desktop rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>(empty list)</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-MaxDesktops<Int32>**

The number of machines from the rule's desktop group to which a user is entitled. Where an entitlement is granted to a user group rather than an individual, the number of machines applies to each member of the user group independently.
Required? | false
--- | ---
Default Value | 1
Accept Pipeline Input? | true (ByPropertyName)

**-PublishedName**<String>

The name of the new machine assignment entitlement as seen by the user, and of the assigned desktop following its usage.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null (dynamically inherited from the desktop group)</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-SecureIcaRequired**<Boolean>

Specifies whether the new desktop rule requires the SecureICA protocol to be used for desktop sessions to machines assigned using the entitlement.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null (dynamically inherited from the desktop group)</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-UUID**<Guid>

An optional GUID for this rule.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>A new GUID is generated if none is supplied.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.
Required?  |  false
--- | ---
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
Accept Pipeline Input? | false

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.AssignmentPolicyRule

New-BrokerAssignmentPolicyRule returns the newly created desktop rule in the assignment policy rule.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $dg = Get-BrokerDesktopGroup 'Sales Support'
C:\PS> New-BrokerAssignmentPolicyRule 'UK Office' -DesktopGroupUId $dg.Uid -IncludedUsers sales\uk-staff -PublishedName 'Sales Desktop'

 Creates a desktop rule in the assignment policy that grants all members of the SALES\uk-staff group an entitlement to a single machine from the Sales Support desktop group. The entitlement name seen by users is Sales Desktop.
New-BrokerCatalog

Oct 10, 2013

Adds a new catalog to the site.

Syntax

New-BrokerCatalog [-Name] <String> [-AllocationType] <AllocationType> [-CatalogKind] <CatalogKind> [-PvsForVM <String>][ -Description <String>][ -IsRemotePC <Boolean>][ -MachinesArePhysical <Boolean>][ -MinimumFunctionalLevel <FunctionalLevel>][ -PvsAddress <String>][ -PvsDomain <String>][ -Scope <String>][ -UUID <Guid>][ -LoggingId <Guid>][ -AdminAddress <String>]

[-CommonParameters-]

New-BrokerCatalog [-Name] <String> [-AllocationType] <AllocationType> [-AllocationType] <AllocationType> [-CatalogKind] <CatalogKind> [-PvsForVM <String>][ -Description <String>][ -IsRemotePC <Boolean>][ -MachinesArePhysical <Boolean>][ -MinimumFunctionalLevel <FunctionalLevel>][ -PvsAddress <String>][ -PvsDomain <String>][ -Scope <String>][ -UUID <Guid>][ -LoggingId <Guid>][ -AdminAddress <String>]

[-CommonParameters-]

Detailed Description

New-BrokerCatalog adds a catalog through which machines can be provided to the site.

In order for a machine to register in a site, the machine must belong to a catalog with which it is compatible. The compatibility of a machine with a catalog is determined by two of the parameters of New-BrokerCatalog:

- MinimalFunctionalLevel: The minimal functional level supported in the catalog. The functional level of the machine is determined by the capabilities of the Citrix VDA software on it.
- SessionSupport: The session support (single/multi) of the catalog. The session support of the machine is determined by the variant of the Citrix VDA software installed (workstation/terminal services, respectively).

Related topics

Get-BrokerCatalog
Rename-BrokerCatalog
Remove-BrokerCatalog
Set-BrokerCatalog

Parameters

- **-Name**: <String>

  Specifies a name for the catalog. Each catalog within a site must have a unique name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-AllocationType**: <AllocationType>

  Specifies how machines in the catalog are assigned to users. Values can be:

  - Static - Machines in a catalog of this type are permanently assigned to a user.
  - Permanent - equivalent to 'Static'.
  - Random - Machines in a catalog of this type are picked at random and temporarily assigned to a user.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-CatalogKind**: <CatalogKind>

  Deprecated: The type of machines the catalog will contain. Values can be: ThinCloned, SingleImage, PowerManaged, Unmanaged, Pvs, Pvd or PvsPvd.

  Thin-Cloned, Single-Image and Personal vDisk Catalogs

  Thin-cloned and single-image catalog kinds are for machines created and managed with Provisioning Services for VMs. All machines in this type of catalog are managed, and so must be associated with a hypervisor connection.

  A thin-cloned catalog is used for original golden VM images that are cloned when they are assigned to a VM, and users’ changes to the VM image are retained after the VM is restarted.

  A single-image catalog is used when multiple machines provisioned with Provisioning Services for VMs all share a single golden VM image when they run and, when restarted, they revert to the original VM image state.

  A personal vDisk catalog is similar to a single-image catalog, but it also uses personal vDisk technology.

PowerManaged
This catalog kind is for managed machines that are manually provisioned by administrators. All machines in this type of catalog are managed, and so must be associated with a hypervisor connection.

Unmanaged

This catalog kind is for unmanaged machines, so there is no associated hypervisor connection.

PVS

This catalog kind is for managed machines that are provisioned using Provisioning Services. All machines in this type of catalog are managed, and so must be associated with a hypervisor connection. Only shared desktops are suitable for this catalog kind.

A Provisioning Services-personal vDisk (PvsPvd) catalog is similar to a Provisioning Services catalog, but it also uses personal vDisk technology.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProvisioningType</td>
<td>true</td>
<td>ProvisioningType</td>
<td>ByPropertyName</td>
</tr>
</tbody>
</table>

Specifies the ProvisioningType for the catalog. Values can be:
- Manual - No provisioning
- PVS - Machine provisioned by PVS (machine may be physical, blade, VM,...).
- MCS - Machine provisioned by MCS (machine must be VM).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>SessionSupport</td>
<td>true</td>
<td>SessionSupport</td>
<td>ByPropertyName</td>
</tr>
</tbody>
</table>

Specifies whether machines in the catalog are single or multi-session capable. Values can be:
- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>PersistUserChanges</td>
<td>true</td>
<td>PersistUserChanges</td>
<td>ByPropertyName</td>
</tr>
</tbody>
</table>

Specifies the location where the user changes will be persisted. Can only be set for single session catalogs. Values can be:
- OnLocal - User changes are persisted locally.
- Discard - User changes are discarded.
- OnPvd - User changes are persisted on the personal vDisk.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>PvsForVM</td>
<td>false</td>
<td>PvsForVM</td>
<td>ByPropertyName</td>
</tr>
</tbody>
</table>

Deprecated:
Identifies the provisioning scheme used by this catalog. To be specified in the format: ProvisioningSchemeGuid:ServiceGroupGuid. Applicable only to thin-cloned, single-image or personal vDisk catalogs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>true</td>
<td>Description</td>
<td>ByPropertyName</td>
</tr>
</tbody>
</table>
A description for the catalog.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsRemotePC</td>
<td>false</td>
<td>false</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-IsRemotePC (Boolean)

Specifies whether this is to be a Remote PC catalog.

IsRemotePC can only be enabled when:
- SessionSupport is SingleSession
- MachinesArePhysical is true.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>MachinesArePhysical</td>
<td>false</td>
<td>false</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-MachinesArePhysical (Boolean)

Specifies whether machines in the catalog can be power-managed by the Citrix Broker Service. Where the Citrix Broker Service cannot control the power state of the machine specify $true, otherwise $false. Can only be specified together with a provisioning type of Pvs or Manual, or if used with the legacy CatalogKind parameter only with Pvs or PvsPvd catalog kinds.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>MinimumFunctionalLevel</td>
<td>false</td>
<td>The FunctionalLevel of the current release (L7)</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-MinimumFunctionalLevel (FunctionalLevel)

The minimum FunctionalLevel required for machines to register in the site.

Valid values are L5, L7

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>PvsAddress</td>
<td>false</td>
<td>The FunctionalLevel of the current release (L7) by default no machines with less than the most current FunctionalLevel will be functional.</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-PvsAddress (String)

Specifies the URL of the Provisioning Services server. Only applicable to Provisioning Services or Provisioning Services-personal vDisk catalogs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>PvsDomain</td>
<td>false</td>
<td>false</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-PvsDomain (String)

Specifies the Active Directory domain of the Provisioning Services server. Only applicable to Provisioning Services or Provisioning Services-personal vDisk catalogs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>false</td>
<td>false</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Scope (String[])

Specifies the name of the delegated administration scope to which the catalog belongs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>UUID</td>
<td>false</td>
<td>false</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-UUID (Guid)

An optional GUID for this catalog.
### -LoggingId\< Guid\>

Specifies the identifier of the high level operation that this cmdlet call forms part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>A new GUID is generated if none is supplied.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -AdminAddress\< String\>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -ProvisioningSchemeId\< Guid\>

Specifies the identity of the MCS provisioning scheme the new catalog is associated with (can only be specified for new catalogs with a ProvisioningType of MCS).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>$null</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

#### Input Type

None. You cannot pipe input into this cmdlet.

#### Return Values

Citrix.Broker.Admin.SDK.Catalog

New-BrokerCatalog returns the created catalog.

#### Examples

```
C:\PS> New-BrokerCatalog -AllocationType Static -CatalogKind Unmanaged -Description "Catalog1 Description" -Name "Catalog1 Name"

This command creates a catalog that can contain unmanaged physical or virtual machines that are permanently assigned to the user.
```

```
C:\PS> New-BrokerCatalog -AllocationType Random -CatalogKind PowerManaged -Description "catalog 2 Description" -Name "Catalog2 Name"

This command creates a catalog that can contain power-managed machines that are randomly assigned to the user.
```

```
C:\PS> New-BrokerCatalog -AllocationType Random -CatalogKind PVS -Description "PVS Catalog Desc" -Name "PVS Catalog Name" -PvsAddress "pvsServer@pvsDomain.com" -PvsDomain "pvsDomain.com"

This command creates a catalog that can contain managed machines that are provisioned using Provisioning Services.
```

New-BrokerConfigurationSlot

Oct 10, 2013
Creates a new configuration slot.

Syntax
New-BrokerConfigurationSlot [-Name] <String> -SettingsGroup <String> [-Description <String>] [-LoggingId <Guid>] [-AdminAddress <String>][<CommonParameters>]

Detailed Description
Creates a new configuration slot. The SettingsGroup of the slot determines the particular collection of related settings that may be specified in a machine configuration associated with this slot.

For example, the configuration slot may be restricted to configuring Citrix User Profile Manager settings by specifying the SettingsGroup parameter as "G=UPM".

Related topics
Get-BrokerConfigurationSlot
Remove-BrokerConfigurationSlot
New-BrokerMachineConfiguration

Parameters
- **-Name**: <String>
  
  Name of the new configuration slot. This must be alphanumeric and not contain white space.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-SettingsGroup**: <String>
  
  The settings group determines the particular collection of related settings that may be controlled by this slot. This must match the format of a Citrix Group Policy configuration group (e.g. "G=UPM"). Only settings that have this exact group may be specified in a machine configuration associated with this configuration slot.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-Description**: <String>
Description of configuration slot.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-LoggingId</Guid>**

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress</String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.ConfigurationSlot

New-BrokerConfigurationSlot returns an object representing the newly created configuration slot

Examples

-------------------------- EXAMPLE 1 --------------------------

New-BrokerConfigurationSlot -Name "UPM" -SettingsGroup "G=UPM"
Create a new slot named "UPM" to configure settings specific to "User Profile Management"
New-BrokerConfiguredFTA

Oct 10, 2013

Creates a file type association with a published application.

Syntax

New-BrokerConfiguredFTA -ImportedFTA <ImportedFTA> -ApplicationUid <Int32> [-UUID <Guid>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

New-BrokerConfiguredFTA -ExtensionName <String> -HandlerName <String> -ApplicationUid <Int32> [-ContentType <String>] [-HandlerDescription <String>] [-HandlerOpenArguments <String>] [-UUID <Guid>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Creates an association between a file type and a published application for the purposes of the content redirection.

File type association associates a file extension (such as "txt") with an application (such as Notepad). In a Citrix environment file type associations on a user device can be configured so that when an user clicks on a document it launches the appropriate published application. This is known as "content redirection".

Configured file type associations are different from imported file type associations. Configured file type associations are those that are actually associated with published applications for the purposes of content redirection. Imported file type associations are lists of known file type associations for a given desktop group. See Update-BrokerImportedFTA for more information about imported file type associations.

This cmdlet has two parameter sets, which correspond to the cmdlet’s two use cases.

The first use case leverages imported file type associations to configure file types for published applications. Information about the file type association is read from the imported object. See the Update-BrokerImportedFTA cmdlet for more information about importing file type associations from a worker machine.

The second use case is more complex and allows you to create your own file type association without having to import it first. This also lets you create custom file type associations that may not already exist on the worker machines. This use case is more error-prone, however, because the individual attributes of the file type association must be correctly specified by you.

Related topics

Get-BrokerImportedFTA
Get-BrokerConfiguredFTA
Remove-BrokerConfiguredFTA

Parameters

-ApplicationUid <Int32>

Specifies the application with which the file type should be associated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-ImportedFTA <ImportedFTA>

Specifies the importedFTA object to use for creating the ConfiguredFTA object. All values needed to create a ConfiguredFTA object are read from the ImportedFTA object.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-ExtensionName <String>

Specifies the extension name for the file type association. For example, "txt" or "doc".

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-HandlerName <String>

Specifies the name of the handler for the file type association (as seen in the Registry). For example, "TXTFILE" or "Word.Document.8".

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-UUID <Guid>

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
An optional GUID for this ConfiguredFTA.

| Required? | false |
| Default Value | A new GUID is generated if none is supplied. |
| Accept Pipeline Input? | false |

-LoggingId<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

-ContentType<String>

Specifies the content type of the file type (as listed in the Registry). For example, content type would be "text/plain" or "application/msword".

| Required? | false |
| Default Value | null |
| Accept Pipeline Input? | true (ByPropertyName) |

-HandlerDescription<String>

Specifies the description of the handler for the file type association.

| Required? | false |
| Default Value | null |
| Accept Pipeline Input? | true (ByPropertyName) |

-HandlerOpenArguments<String>

Specifies the arguments for the open command that the handler should use. For example, "%1".

| Required? | false |
| Default Value | null |
| Accept Pipeline Input? | true (ByPropertyName) |

Input Type

Variable, based on property name. This cmdlet does accept input from the pipeline but only by property name.

Return Values

Citrix.Broker.Admin.SDK.ConfiguredFTA

This cmdlet returns a single ConfiguredFTA object.

Examples

------------------------------- EXAMPLE 1 -----------------------------

C:\PS> $app = Get-BrokerApplication 'Notepad'
C:\PS> $fta = Get-BrokerImportedFTA -ExtensionName '.txt'
C:\PS> New-BrokerConfiguredFTA -ImportedFTA $fta -ApplicationUid $app.Uid

Gets the Uid for the application, gets the ImportedFTA object for the file extension, and finally associates "*.txt" with the published "Notepad" application.

Note that the Get-BrokerImportedFTA cmdlet may return more than one ImportedFTA objects for a specific extension name. See the help for that cmdlet for more details.

------------------------------- EXAMPLE 2 -----------------------------
C:\PS> $app = Get-BrokerApplication "Notepad"
C:\PS> New-BrokerConfiguredFTA -ApplicationUid $app.Uid -ExtensionName "txt" -HandlerName "txtfile" -ContentType "text/plain" -HandlerDescription "Text Document" -HandlerOpenArguments "%1"

This example is identical to the first, but shows the second use case of the cmdlet, specifying each attribute manually.
New-BrokerDelayedHostingPowerAction

Oct 10, 2013
Causes a power action to be queued after a delay.

Syntax


Detailed Description

Causes a power action to be queued after the specified period of time.

Only ShutDown or Suspend actions can be requested to be delayed in this manner.

For a detailed description of the queuing mechanism, see 'help about_Broker_PowerManagement'.

Related topics

Get-BrokerDelayedHostingPowerAction

New-BrokerDelayedHostingPowerAction

Parameters

-MachineName <String>

Specifies the machine that the action is to be performed on.

The machine can be identified by DNS name, short name, SID, or name of the form domain\machine.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByPropertyName) |

-Action <PowerManagementAction>

Specifies the power state change action that is to be performed on the specified machine after the specified delay.

Valid values are Shutdown and Suspend.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByPropertyName) |

-Delay <TimeSpan>
Specifies a timespan delay before the action is queued.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.DelayedHostingPowerAction

New-BrokerDelayedHostingPowerAction returns the created delayed power action.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> New-BrokerDelayedHostingPowerAction -Action Shutdown -MachineName 'XD_VDA1' -Delay '00:02:00'
Causes the machine called XD_VDA1 to be shut down after a delay of two minutes.
New-BrokerDesktopGroup

Oct 10, 2013
Create a new desktop group for managing the brokering of groups of desktops.

Syntax


Detailed Description

The New-BrokerDesktopGroup cmdlet creates a new broker desktop group that can then be used to manage the brokering settings of all desktops within that desktop group. Once the desktop group has been created, you can create desktops in it by adding the appropriate broker machines to it using the Add-BrokerMachine or Add-BrokerMachinesToDesktopGroup cmdlets.

Desktop groups hold settings that apply to all desktops they contain.

For any automatic power management settings of a desktop group to take effect, the group's TimeZone property must be specified. Automatic power management operations include pool management (power time schemes), reboot schedules, session disconnect and logoff actions, and powering on assigned machines etc.

Related topics

Get-BrokerDesktopGroup
Set-BrokerDesktopGroup
Rename-BrokerDesktopGroup
Remove-BrokerDesktopGroup
Add-BrokerMachine
Add-BrokerMachinesToDesktopGroup
Get-BrokerSite

Parameters

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1143
### -Name<String>

The name of the new broker desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -DesktopKind<DesktopKind>

The kind of desktops this group will hold. Valid values are Private and Shared.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -AutomaticPowerOnForAssigned<Boolean>

Specifies whether assigned desktops in the desktop group should be automatically started at the start of peak time periods. Only relevant for groups whose DesktopKind is Private.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>true</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -AutomaticPowerOnForAssignedDuringPeak<Boolean>

Specifies whether assigned desktops in the desktop group should be automatically started throughout peak time periods. Only relevant for groups whose DesktopKind is Private and which have AutomaticPowerOnForAssigned set to true.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -ColorDepth<ColorDepth>

Specifies the color depth that the ICA session should use for desktops in this group. Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>TwentyFourBit</td>
</tr>
</tbody>
</table>
## Accept Pipeline Input?
```
true (ByPropertyName)
```

### DeliveryType <DeliveryType>

Specifies whether desktops, applications, or both, can be delivered from machines contained within the new desktop group. Desktop groups with a DesktopKind of Private cannot be used to deliver both desktops and applications. Defaults to DesktopsOnly if not specified.

Valid values are DesktopsOnly, AppsOnly, and DesktopsAndApps.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>DesktopsOnly</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### Description <String>

A description for this desktop group useful for administrators of the site.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### Enabled <Boolean>

Whether the desktop group should be in the enabled state; disabled desktop groups do not appear to users.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>true</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### IconUid <Int32>

The UID of the broker icon to be displayed to users for their desktop(s) in this desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The Uid of the default desktop icon in this site - use the Get-BrokerSite cmdlet to find this value.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### InMaintenanceMode <Boolean>

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
Whether the desktop should be created in maintenance mode; a desktop group in maintenance mode will not allow users to connect or reconnect to their desktops.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **IsRemotePC**<Boolean>

Specifies whether this is to be a Remote PC desktop group.

IsRemotePC can only be enabled when:
- SessionSupport is SingleSession
- DeliveryType is DesksopOnly
- DesktopKind is Private

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **MinimumFunctionalLevel**<FunctionalLevel>

The minimum FunctionalLevel required for machines to work successfully in the desktop group.

Valid values are L5, L7

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The FunctionalLevel of the current release (L7); by default no machines with less than the most current FunctionalLevel will be functional.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **OffPeakBufferSizePercent**<Int32>

The percentage of machines in the desktop group that should be kept available in an idle state outside peak hours.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
-**OffPeakDisconnectAction**<SessionChangeHostingAction>

The action to be performed after a configurable period of a user session disconnecting outside peak hours. Possible values are Nothing, Suspend, or Shutdown.

| Required? | false |
| Default Value | Nothing |
| Accept Pipeline Input? | true (ByPropertyName) |

-**OffPeakDisconnectTimeout**<Int32>

The number of minutes before the configured action should be performed after a user session disconnects outside peak hours.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | true (ByPropertyName) |

-**OffPeakExtendedDisconnectAction**<SessionChangeHostingAction>

The action to be performed after a second configurable period of a user session disconnecting outside peak hours. Possible values are Nothing, Suspend, or Shutdown.

| Required? | false |
| Default Value | Nothing |
| Accept Pipeline Input? | true (ByPropertyName) |

-**OffPeakExtendedDisconnectTimeout**<Int32>

The number of minutes before the second configured action should be performed after a user session disconnects outside peak hours.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | true (ByPropertyName) |

-**OffPeakLogOffAction**<SessionChangeHostingAction>

The action to be performed after a configurable period of a user session ending outside peak hours. Possible values are Nothing, Suspend, or Shutdown.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Nothing</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-OffPeakLogOffTimeout**<Int32>

The number of minutes before the configured action should be performed after a user session ends outside peak hours.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-PeakBufferSizePercent**<Int32>

The percentage of machines in the desktop group that should be kept available in an idle state in peak hours.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-PeakDisconnectAction**<SessionChangeHostingAction>

The action to be performed after a configurable period of a user session disconnecting in peak hours. Possible values are Nothing, Suspend, or Shutdown.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Nothing</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-PeakDisconnectTimeout**<Int32>

The number of minutes before the configured action should be performed after a user session disconnects in peak hours.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-PeakExtendedDisconnectAction**<SessionChangeHostingAction>

The action to be performed after a second configurable period of a user session disconnecting in peak hours. Possible
values are Nothing, Suspend, or Shutdown.

| Required?   | false                          |
| Default Value | Nothing                      |
| Accept Pipeline Input? | true (ByPropertyName) |

**-PeakExtendedDisconnectTimeout<Int32>**

The number of minutes before the second configured action should be performed after a user session disconnects in peak hours.

| Required?   | false                          |
| Default Value | 0                            |
| Accept Pipeline Input? | true (ByPropertyName) |

**-PeakLogOffAction<SessionChangeHostingAction>**

The action to be performed after a configurable period of a user session ending in peak hours. Possible values are Nothing, Suspend, or Shutdown.

| Required?   | false                          |
| Default Value | Nothing                      |
| Accept Pipeline Input? | true (ByPropertyName) |

**-PeakLogOffTimeout<Int32>**

The number of minutes before the configured action should be performed after a user session ends in peak hours.

| Required?   | false                          |
| Default Value | 0                            |
| Accept Pipeline Input? | true (ByPropertyName) |

**-ProtocolPriority<String[]>**

A list of protocol names in the order in which they should be attempted for use during connection.

| Required?   | false                          |
| Default Value | null                         |
| Accept Pipeline Input? | true (ByPropertyName) |
- **PublishedName**: `<String>`
  
  The name that will be displayed to users for their desktop(s) in this desktop group.

  | Required? | false |
  | Default Value | The same value as that supplied for the name of the desktop group. |
  | Accept Pipeline Input? | true (ByPropertyName) |

- **Scope**: `<String[]>`
  
  Specifies the name of the delegated administration scope to which the desktop group should belong.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | true (ByPropertyName) |

- **SecureIcaRequired**: `<Boolean>`
  
  Whether HDX connections to desktops in the new desktop group require the use of a secure protocol.

  | Required? | false |
  | Default Value | false |
  | Accept Pipeline Input? | true (ByPropertyName) |

- **SessionSupport**: `<SessionSupport>`
  
  Specifies whether machines in the desktop group are single or multi-session capable. Values can be:
  
  - **SingleSession** - Single-session only machine.
  - **MultiSession** - Multi-session capable machine.

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | true (ByPropertyName) |

- **ShutdownDesktopsAfterUse**: `<Boolean>`
  
  Whether desktops in this desktop group should be automatically shut down when each user session completes (only relevant to power-managed desktops).

  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | |
- **TimeZone**: `String`

The time zone in which this desktop group's machines reside.

The time zone must be specified for any of the group's automatic power management settings to take effect. Automatic power management operations include pool management (power time schemes), reboot schedules, session disconnect and logoff actions, and powering on assigned machines etc.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **TurnOnAddedMachine**: `Boolean`

This flag specifies whether the Broker Service should attempt to power on machines when they are added to the desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>$false for single session machines and $true for multi-session machines.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **UUID**: `Guid`

An optional GUID for this desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>A new GUID is generated if none is supplied.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **LoggingId**: `Guid`

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.DesktopGroup

The newly created desktop group.

Notes

Once a new desktop group is created, you can create desktops in it by adding the appropriate broker machines to it using the Add-BrokerMachine or Add-BrokerMachinesToDesktopGroup cmdlets.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> New-BrokerDesktopGroup "Assigned Desktops" -PublishedName "MyDesktop" -DesktopKind Private
Create a desktop group to manage the brokering of private desktops, which will appear to users with the name "MyDesktop".
New-BrokerEntitlementPolicyRule

Oct 10, 2013
Creates a new desktop rule in the site's entitlement policy.

Syntax

IncludedUsers <User[]>] [-PublishedName <String>] [-SecureIcaRequired <Boolean>] [-UUID <Guid>] [-LoggingId <Guid>] [-AdminAddress <String>]
[<CommonParameters>]

Detailed Description

The New-BrokerEntitlementPolicyRule cmdlet adds a new desktop rule to the site's entitlement policy.

A desktop rule in the entitlement policy defines the users who are allowed per-session access to a machine from the rule's associated desktop
group to run a full desktop session.

The following constraints apply when creating a desktop entitlement rule for a desktop group:

- The group's desktop kind must be Shared
- The group's delivery type must be DesktopsOnly or DesktopsAndApps

When a user selects a desktop entitlement published from a shared group, a machine is selected from the group on which to run the desktop
session. No permanent association exists between the user and the selected machine; once the session ends the association also ends.

Multiple desktop rules in the entitlement policy can apply to the same desktop group. Where a user is granted an entitlement by more than one
rule for the same group, they can use as many desktop sessions at the same time as they have entitlements.

Related topics

Get-BrokerEntitlementPolicyRule
Set-BrokerEntitlementPolicyRule
Rename-BrokerEntitlementPolicyRule
Remove-BrokerEntitlementPolicyRule

Parameters

- **-Name <String>**

  Specifies the administrative name of the new desktop rule. Each rule in the site's entitlement policy must have a unique name (irrespective of
whether they are desktop or application rules).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

| Accept Pipeline Input? | true (ByPropertyName) |

- **-DesktopGroupUid <Int32>**

  Specifies the unique ID of the desktop group to which the new desktop rule applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-ColorDepth</code></td>
<td>Specifies the color depth of any desktop sessions launched by a user from this entitlement.</td>
</tr>
<tr>
<td></td>
<td>Valid values are $null$, FourBit, EightBit, SixteenBit, and TwentyFourBit.</td>
</tr>
<tr>
<td></td>
<td>The default null value indicates that the equivalent setting from the rule's desktop group is used.</td>
</tr>
</tbody>
</table>

| Required?          | false                                                                 |
| Default Value      | null (dynamically inherited from the desktop group)                 |
| Accept Pipeline Input? | true (ByPropertyName)        |

| `-Description`     | Specifies an optional description of the new desktop rule. The text may be visible to the end user, for example, as a tooltip associated with the desktop entitlement. |
|                    | The default null value indicates that the equivalent setting from the rule's desktop group is used. |

| Required?          | false                                                                 |
| Default Value      | null (dynamically inherited from the desktop group)                 |
| Accept Pipeline Input? | true (ByPropertyName)        |

| `-Enabled`         | Specifies whether the new desktop rule is initially enabled. A disabled rule is ignored when evaluating the site's entitlement policy. |

| Required?          | false                                                                 |
| Default Value      | true                                                                  |
| Accept Pipeline Input? | true (ByPropertyName)        |

| `-ExcludedUserFilterEnabled` | Specifies whether the excluded users filter is initially enabled. If the filter is disabled then any user entries in the filter are ignored when entitlement policy rules are evaluated. |

| Required?          | false                                                                 |
| Default Value      | false                                                                 |
| Accept Pipeline Input? | true (ByPropertyName)        |

| `-ExcludedUsers`   | Specifies the excluded users filter of the desktop rule, that is, the users and groups who are explicitly denied an entitlement to a desktop session from the new rule. |
|                    | This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter. |

| Required?          | false                                                                 |
| Default Value      | (empty list)                                                          |

---

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1154
### Accept Pipeline Input?

| Accept Pipeline Input? | true (ByPropertyName) |

### IconUid<br>&lt;Int32&gt;

Specifies the unique ID of the icon used to display the desktop session entitlement to the user.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

| Required? | false |
| Default Value | null (dynamically inherited from the desktop group) |
| Accept Pipeline Input? | true (ByPropertyName) |

### IncludedUserFilterEnabled<br>&lt;Boolean&gt;

Specifies whether the included users filter is initially enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a desktop session by the new rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

| Required? | false |
| Default Value | true |
| Accept Pipeline Input? | true (ByPropertyName) |

### IncludedUsers<br>&lt;User[]&gt;

Specifies the included users filter of the rule, that is, the users and groups who are granted an entitlement to a desktop session by the new rule.

If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

| Required? | false |
| Default Value | (empty list) |
| Accept Pipeline Input? | true (ByPropertyName) |

### PublishedName<br>&lt;String&gt;

The name of the new desktop session entitlement as seen by the user.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

| Required? | false |
| Default Value | null (dynamically inherited from the desktop group) |
| Accept Pipeline Input? | true (ByPropertyName) |

### SecureIcaRequired<br>&lt;Boolean&gt;

Specifies whether the new desktop rule requires the SecureICA protocol for desktop sessions launched using the entitlement.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

| Required? | false |
Default Value | null (dynamically inherited from the desktop group)  
Accept Pipeline Input? | true (ByPropertyName)  

-UUID<Guid>

An optional GUID for this rule.

| Required? | false  
| Default Value | A new GUID is generated if none is supplied.  
| Accept Pipeline Input? | true (ByPropertyName)  

-LoggingId<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false  
| Default Value |  
| Accept Pipeline Input? | false  

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false  
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.  
| Accept Pipeline Input? | false  

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.EntitlementPolicyRule

New-BrokerEntitlementPolicyRule returns the newly created desktop rule in the entitlement policy.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $dg = Get-BrokerDesktopGroup 'Customer Support'
C:\PS> New-BrokerEntitlementPolicyRule 'UK Office'-DesktopGroupId $dg.Uid -IncludedUsers support\uk-staff -PublishedName 'Support Desktop'

Creates an desktop rule in the entitlement policy that entitles all members of the SUPPROT\uk-staff group to a desktop session from the Customer Support desktop group. The desktop entitlement name seen by users is Support Desktop.
New-BrokerHostingPowerAction

Oct 10, 2013
Creates a new action in the power action queue.

Syntax


Detailed Description

The New-BrokerHostingPowerAction cmdlet adds a new power action record into the queue of power actions to be performed. The power actions in the queue are processed on a priority basis and sent to the relevant hypervisor to change the power state of a virtual machine.

A power action record defines the action to be performed, the machine on which the action is to be performed, and an initial priority value for the action. Multiple actions may be created that relate to the same machine.

For a detailed description of the queuing mechanism, see 'help about_Broker_PowerManagement'.

Related topics

Get-BrokerHostingPowerAction
Set-BrokerHostingPowerAction
Remove-BrokerHostingPowerAction

Parameters

-MachineName <String>
Specifies the machine that the action is to be performed on.

The machine can be identified by DNS name or short name or SID or 'machine\domain' form name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Action <PowerManagementAction>

Specifies the power state change action that is to be performed on the specified machine.

Valid values are: TurnOn, TurnOff, ShutDown, Reset, Restart, Suspend and Resume.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | true (ByPropertyName)

- **ActualPriority**<Int32>

Specifies an initial priority value for the action in the queue.

This priority is the current action priority; the 'base' priority for actions created via this cmdlet is always 30. Numerically lower priority values indicate more important actions that are processed in preference to actions with numerically higher priority settings.

| Required? | false |
| Default Value | 30 |

Accept Pipeline Input? | true (ByPropertyName)

- **LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value |  |

Accept Pipeline Input? | false

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |

Accept Pipeline Input? | false

Input Type
None You cannot pipe input into this cmdlet.

Return Values
Citrix.BrokerAdmin.SDK.HostingPowerAction

New-BrokerHostingPowerAction returns the newly created power action record.

Examples
EXAMPLE 1

C:\PS> New-BrokerHostingPowerAction -Action Shutdown -MachineName 'XD_VDA1'
Causes the machine called 'XD_VDA1' to be shut down.
New-BrokerHypervisorConnection

Oct 10, 2013
Creates a new hypervisor connection.

Syntax


Detailed Description

The New-BrokerHypervisorConnection cmdlet creates a new hypervisor connection.

Related topics

Get-BrokerHypervisorConnection
Remove-BrokerHypervisorConnection
Set-BrokerHypervisorConnection

Parameters

-HypHypervisorConnectionUid <Guid>
The GUID that identifies the hypervisor connection, as defined in DUM.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-PreferredController <String>
The preferred controller machine for the hypervisor connection. Can be specified as (first match is used):

- Full SAM name.
- Full DNS name.
- SID value.
- NetBIOS name (SAM without domain).
- Partial DNS name (DNS name without some or all domain information).

Where not specified, the system selects preferred controller machine based on loading.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>
Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1160
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type
None

Return Values
C:itrix.Broker.Admin.SDK.HypervisorConnection

New-BrokerHypervisorConnection returns an opaque object containing information about the hypervisor connection.

Examples
----------------------------------- EXAMPLE 1 -----------------------------------
C:\PS> New-BrokerHypervisorConnection -PreferredController "domainName\controllerName" -HypHypervisorConnectionUid "d16f4e56-b85e-4ba6-b745-0e978ae4f192"
This command creates a new hypervisor connection with a preferred controller.

----------------------------------- EXAMPLE 2 -----------------------------------
C:\PS> New-BrokerHypervisorConnection -HypHypervisorConnectionUid "d16f4e56-b85e-4ba6-b745-0e978ae4f192"
This command creates a new hypervisor connection, and leaves it to the system to select a preferred controller.
New-BrokerIcon

Oct 10, 2013
Creates a new icon.

Syntax

New-BrokerIcon [-EncodedIconData <String>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Accepts Base64 encoded .ICO format icon data, stores it in the database and returns an Icon object containing theUid
assigned to it.

New-BrokerIcon can be used with the Get-CtxIcon cmdlet from Citrix.Common.Commands, to obtain the Base64 icon. See
Examples for a demonstration.

Related topics

Get-CtxIcon

Get-BrokerIcon

Remove-BrokerIcon

Parameters

-**EncodedIconData**<String>

Specifies the Base64 encoded .ICO format icon data.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop
Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level
Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

None Input cannot be piped to this cmdlet.

**Return Values**

Citrix.BrokerAdmin.SDK.Icon

Returns an Icon object.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Add-PSSnapin Citrix.Common.Commands
C:\PS> $ctxIcon = Get-CtxIcon -FileName C:\Windows\System32\notepad.exe -index 0
C:\PS> $brokerIcon = New-BrokerIcon -EncodedIconData $ctxIcon.EncodedIconData
C:\PS> $desktopGroup = Get-BrokerDesktopGroup -Name 'MyDesktopGroup'
C:\PS> Set-BrokerDesktopGroup $desktopGroup -IconUid $brokerIcon.Uid

Extracts the first icon resource from notepad.exe, and sets this as the icon for a desktop group.
New-BrokerMachine

Oct 10, 2013
Adds a machine that can be used to run desktops and applications.

Syntax

Detailed Description
By adding a machine to a catalog, New-BrokerMachine adds a machine to the site, and is the first step in making the machine available to run users' desktops and applications. The machine may be physical or virtual.

For physical machines, you must specify the machine's SID and the catalog to which it will belong. For virtual machines which are not provisioned by MCS, you must also provide the hypervisor connection responsible for running the machine and the hosted machine ID by which the hypervisor recognizes the machine.

The machine must support the expected capabilities of the catalog: the catalog specifies a SessionType and a MinimalFunctionalLevel. The session support of the machine is determined by the type of Citrix VDA software installed (server or workstation) and the functional level depends on the version of the Citrix VDA software installed. The New-BrokerMachine command will complete successfully if these are not correct but the machine will be unable to register.

For more information about machines, see about_Broker_Machines.

Related topics
Add-BrokerMachine

Parameters

-MachineName <String>
Specify the name of the machine to create (in the form 'domain\machine'). A SID can also be specified.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-CatalogUid <Int32>
The catalog to which this machine will belong.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-AssignedClientName <String>
The client name to which this machine will be assigned. Machines can be assigned to multiple users, a single client IP address, or a single client name, but only to one of these categories at a time.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-AssignedIPAddress <String>
The client IP address to which this machine will be assigned. Machines can be assigned to multiple users, a single client IP address, or a single client name, but only to one of these categories at a time.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>-HostedMachineId</td>
<td>The unique ID by which the hypervisor recognizes the machine. Omit this for physical machines or MCS-provisioned VMs.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>null</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
<td></td>
</tr>
<tr>
<td>-HypervisorConnectionUid</td>
<td>The hypervisor connection that runs the machine. Omit this for physical machines or MCS-provisioned VMs.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>null</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
<td></td>
</tr>
<tr>
<td>-InMaintenanceMode</td>
<td>Specifies whether the machine is initially in maintenance mode. A machine in maintenance mode is not available for new sessions, and for managed machines all automatic power management is disabled.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
<td></td>
</tr>
<tr>
<td>-UUID</td>
<td>An optional GUID for this machine.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>A new GUID is generated if none is supplied.</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
<td></td>
</tr>
<tr>
<td>-LoggingId</td>
<td>Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>-AdminAddress</td>
<td>Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
None You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.Machine

New-BrokerMachine returns the created machine.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> New-BrokerMachine -CatalogUid 2 -MachineName 'domain\machine'
This adds the physical machine with the specified SAM name to this site and places it in the specified catalog.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> New-BrokerMachine -CatalogUid 2 -MachineName 'S-1-5-12-1234567890-1234567890-1234567890-1234'
This adds the physical machine with the specified SID to this site and places it in the specified catalog.

-------------------------- EXAMPLE 3 --------------------------
C:\PS> New-BrokerMachine -CatalogUid 2 -MachineName 'domain\machine' -HostedMachineId 'F8143B4F-7371-4efa-868A-54787EF9F64E' -HypervisorConnectionUid 5
This adds the virtual machine, running on the specified hypervisor, to this site and places it in the catalog.

-------------------------- EXAMPLE 4 --------------------------
C:\PS> $m = New-BrokerMachine -CatalogUid 2 -MachineName 'domain\machine'
C:\PS> Add-BrokerMachine -InputObject $m -DesktopGroup 3
This adds the specified physical machine to the site and uses Add-BrokerMachine to add it to a desktop group.
New-BrokerMachineCommand

Oct 10, 2013
Creates a new command to deliver to a desktop.

Syntax


New-BrokerMachineCommand -SessionUid <Int64> -Category <String> -CommandName <String> [-SendTrigger <MachineCommandTrigger>] [-SendDeadline <TimeSpan>] [-CommandData <Byte[]>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

New-BrokerMachineCommand -Synchronous -MachineUid <Int32> -Category <String> -CommandName <String> [-CommandData <Byte[]>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Create a new command queued for delivery to a desktop. Commands are sent to a specific handler installed on the desktop using the Category parameter. Each handler has its own list of commands identified by the CommandName parameter. Optional command data can be provided using the CommandData parameter in a format specified by the handler.

Commands are targeted at a specific user, session or machine. Commands targeted at a user can be further be restricted to one or more desktop groups.

The SendTrigger is used to restrict the command to a specific event related to the target. For example, when the target machine registers or when the target user reconnects to a session. The command will be sent to the machine when the SendTrigger occurs for the target.

If the Synchronous switch is provided, the target must be a machine and no SendTrigger can be specified. The command is sent immediately to the machine if it is currently registered and fails if the machine is not registered.

Note that the combined length of the Category and CommandName is limited to 64 characters. The Category and CommandName must both be entirely alphanumeric and not include any white space.

Related topics

Get-BrokerMachineCommand
Remove-BrokerMachineCommand

Parameters

-Category <String>
The service on the desktop to send the command to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
**-CommandName**<String>

The name of the command to send (as defined by the service).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-User**<String>

User whose desktop or session should receive the command.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Any user.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-MachineUid**<Int32>

Specific machine that should receive the command.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-SessionUid**<Int64>

Currently logged on user session that should receive the command.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Any session.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Synchronous**<SwitchParameter>

Send the command immediately and block while waiting for the reply.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-CommandData**<Byte[]>

Optional additional data to include with the command.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
### Default Value

<table>
<thead>
<tr>
<th>Field</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Accept Pipeline Input?

<table>
<thead>
<tr>
<th>Field</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true (ByPropertyName)</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -LoggingId<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Field</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Field</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -DesktopGroups<DesktopGroup[]>

Further restrict the command targeted at a user to machines in these desktop groups.

<table>
<thead>
<tr>
<th>Field</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>No restriction by desktop group.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -SendTrigger<MachineCommandTrigger>

Queue command for delivery until this particular event occurs. Valid values are NextContact, Broker, LogOn, Logoff, Disconnect and Reconnect.

<table>
<thead>
<tr>
<th>Field</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>Default value is 'NextContact' so the command is sent during the next communication with the desktop.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -SendDeadline<TimeSpan>

Automatically cancel the command if it not delivered before the specified time span passes.

<table>
<thead>
<tr>
<th>Field</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>Command expires after 24 hours.</td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | true (ByPropertyName)

Input Type

None No parameter is accepted from the input pipeline.

Return Values

Citrix.BrokerAdmin.SDK.MachineCommand

New command that was added to the command queue.Citrix.BrokerAdmin.SDK.MachineSynchronousCommandResponse

When the Synchronous option is used, the command is immediately sent to the specified machine and processed. The SDK object returned describes the command and the result of this command processing.

Notes

Commands are subject to delegated administration restrictions based on the desktop group, category and command name.

Examples

-------------------------- EXAMPLE 1 --------------------------

New-BrokerMachineCommand -Category "UPM" -CommandName "ResetProfile" -DesktopGroups 1 -UserUid 23 -SendTrigger Authentication
Instruct the User Profile Manager service to execute the "ResetProfile" command when user 23 logs on to any machine in desktop group 1

-------------------------- EXAMPLE 2 --------------------------

New-BrokerMachineCommand -Synchronous -Category "MonitorService" -CommandName "EnableLogging" -MachineUid 13
Instruct the monitor service to immediately execute the "EnableLogging" command on the machine having Uid 13.
New-BrokerMachineConfiguration

Oct 10, 2013
Creates a new machine configuration associated with an existing configuration slot.

Syntax


Detailed Description

Creates a new machine configuration containing settings that match the SettingsGroup of the associated configuration slot. This machine configuration can then be applied to a desktop group to have the settings applied to machines in that group.

The SettingsGroup of the configuration slot restricts the permitted settings. Use the SDK snap-in that matches the SettingsGroup to create the encoded settings data.

Related topics

Get-BrokerMachineConfiguration
Set-BrokerMachineConfiguration
Rename-BrokerMachineConfiguration
Remove-BrokerMachineConfiguration
Add-BrokerMachineConfiguration

Parameters

-ConfigurationSlotUid <Int32>
Unique identifier of the configuration slot to associate with this machine configuration.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>None</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-LeafName <String>
Name of the new machine configuration. This must be unique amongst the machine configurations associated with the same configuration slot.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>None</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Policy <Byte[]>
A binary array of encoded settings (policy) data created with the SDK snap-in that matches the SettingsGroup of the configuration slot.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>None</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Description <String>
Description of the new machine configuration.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>None</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type
None
You cannot pipe input into this cmdlet.

Return Values
Citrix.BrokerAdmin.SDK.MachineConfiguration
New-BrokerMachineConfiguration returns the newly created configuration

Notes
Delegated Administration can be used to restrict the configuration slots that an administrator can use and hence which components of the system that can be configured.

Examples
-------------------------- EXAMPLE 1 --------------------------
New-BrokerMachineConfiguration -LeafName "Finance Department" -Description "Finance Dept. User Profile Management policy" -Policy $policy -ConfigurationSlotUid $csUid
Creates a new configuration named "%SlotName%\Finance Department" where %SlotName% is the name of the configuration slot having the Uid $csUid. The encoded settings in the $policy variable must match the SettingsGroup of the configuration slot having the Uid $csUid.
New-BrokerPowerTimeScheme

Oct 10, 2013
Creates a new power time scheme for a desktop group.

Syntax


Detailed Description

The New-BrokerPowerTimeScheme cmdlet adds a new power time scheme to be associated with a desktop group. The power time scheme must relate to days of the week that are not already covered by an existing power time scheme.

Each power time scheme is associated with a particular desktop group, and covers one or more days of the week, defining which hours of those days are considered peak times and which are off-peak times. In addition, the time scheme defines a pool size value for each hour of the day for the days of the week covered by the time scheme. No one desktop group can be associated with two or more time schemes that cover the same day of the week.

See 'help about_Broker_PowerManagement' for a detailed description of the power policy mechanism and pool size management.

Related topics

Get-BrokerPowerTimeScheme
Set-BrokerPowerTimeScheme
Remove-BrokerPowerTimeScheme
Rename-BrokerPowerTimeScheme

Parameters

-Name <String>

Specifies the administrative name of the new power time scheme. Each scheme must have a name which is unique within the site.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-DaysOfWeek <TimeSchemeDays>

Specifies the pattern of days of the week that the power time scheme covers.

Valid values are (singly or a list of) Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday, Weekdays and Weekend.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-DesktopGroupUid <Int32>

Specifies the desktop group that the power time scheme applies to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-DisplayName <String>

Specifies the name of the new power time scheme as displayed in the DesktopStudio console. Each scheme associated with a desktop group must have a display name which is unique within its desktop group, although the same display name can be used on power schemes for different desktop groups.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
Default Value | Accept Pipeline Input?
---|---
Accept Pipeline Input? | true (ByPropertyName)

-PeakHours<Boolean[]>-
A set of 24 boolean flag values, one for each hour of the day. The first value in the array relates to midnight to 00:59, the next one to 1 AM to 01:59 and so on, with the last array element relating to 11 PM to 11:59. If the flag is $true it means that the associated hour of the day is considered a peak time; if $false it means that it is considered off-peak.

Required? | false
Default Value | 24 $false values, meaning all hours are off-peak
Accept Pipeline Input? | true (ByPropertyName)

-PoolSize<int32[]>-
A set of 24 integer values, one for each hour of the day. The first value in the array relates to midnight to 00:59, the next one to 1 AM to 01:59 and so on, with the last array element relating to 11 PM to 11:59. The value defines the number of machines (either as an absolute number or a percentage of the machines in the desktop group) that are to be maintained in a running state, whether they are in use or not. A value of -1 has special meaning: pool size management does not apply during such hours.

Required? | false
Default Value | 24 values of '-1', meaning no pool size management is to be performed
Accept Pipeline Input? | true (ByPropertyName)

-PoolUsingPercentage<Boolean>
A boolean flag to indicate whether the integer values in the pool size array are to be treated as absolute values (if this value is $false) or as percentages of the number of machines in the desktop group (if this value is $true).

Required? | false
Default Value | false
Accept Pipeline Input? | true (ByPropertyName)

-LoggingId<Guid>
Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Required? | false
Default Value | 
Accept Pipeline Input? | true (ByPropertyName)

-AdminAddress<String>
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

Required? | false
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
Accept Pipeline Input? | false

Input Type
None You cannot pipe input into this cmdlet.

Return Values
Citrix.BrokerAdminSDK.PowerTimeScheme

New-BrokerPowerTimeScheme returns the newly created power time scheme.
Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> New-BrokerPowerTimeScheme -Name 'First Half Week' -DaysOfWeek Weekend,Monday,Tuesday -DesktopGroupUid 3 -PeakHours (0..23 | %{ $_ -gt 8 -and $_ -lt 18 } )

Creates a new scheme attached to the desktop group whose UID value is 3. This new scheme covers the weekend and Monday and Tuesday, and defines 'peak' hours as 9am to 17:59, with all other times being 'off-peak'. No pool size values are supplied, so all size values for all the hours default to -1.
New-BrokerRebootSchedule

Oct 10, 2013

Creates a new reboot schedule for a desktop group.

Syntax


Detailed Description

The New-BrokerRebootSchedule cmdlet is used to define a reboot schedule for a desktop group.

Related topics

Get-BrokerRebootSchedule
Set-BrokerRebootSchedule
Remove-BrokerRebootSchedule
Start-BrokerRebootCycle

Parameters

-DesktopGroupName <String>
The name of the desktop group that this reboot schedule is applied to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-RebootDuration <Int32>
Approximate maximum number of minutes over which the scheduled reboot cycle runs.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-DesktopGroupUid <Int32>
The Uid of the desktop group that this reboot schedule is applied to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Day <RebootScheduleDays>
For weekly schedules, the day of the week on which the scheduled reboot cycle starts (one of Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Enabled <Boolean>
Boolean that indicates if the new reboot schedule is enabled.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Frequency <RebootScheduleFrequency>
Frequency with which this schedule runs (either Weekly or Daily).
### Required?
- *false*

### Default Value
- *false*

### Accept Pipeline Input?
- true (ByPropertyName)

### -StartTime
**Time of day at which the scheduled reboot cycle starts (HH:MM)**

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -WarningDuration
**Time prior to the initiation of a machine reboot at which warning message is displayed in all user sessions on that machine. If the warning duration is zero then no message is displayed.**

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -WarningMessage
**Warning message displayed in user sessions on a machine scheduled for reboot. If the message is blank then no message is displayed.**

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -WarningTitle
**The window title used when showing the warning message in user sessions on a machine scheduled for reboot.**

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -LoggingId
**Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.**

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -AdminAddress
**Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.**

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### Notes
- Input Type: None
- Input cannot be piped to this cmdlet.
- Return Values: Citrix.Broker.Admin.SDK.RebootSchedule
- Examples:

```
C:\PS> New-BrokerRebootSchedule -DesktopGroupName BankTellers -Frequency Daily -StartTime "02:00" -Enabled $true -Duration 120
Schedules the machines in the desktop group named 'BankTellers' to be rebooted every night between 2 AM and 4 AM.
```

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1177
EXAMPLE 2

C:\PS> New-BrokerRebootSchedule -DesktopGroupUid 17 -Frequency Weekly -Day Saturday -Start_Time "01:00" -Enabled $true -Duration 240 -WarningTitle "WARNING: Reboot pending" -WarningMessage

Schedules the machines in the desktop group having Uid 17 to be rebooted every Saturday night between 1 AM and 5 AM. Ten minutes prior to rebooting, each machine will display a message box with the title "WARNING: Reboot pending" and message "Save your work" in every user session.
New-BrokerRemotePCAccount

Oct 10, 2013
Create a new RemotePCAccount.

Syntax
New-BrokerRemotePCAccount -CatalogUid <Int32> -OU <String> [-AllowSubfolderMatches <Boolean>] [-MachinesExcluded <String[]>][-MachinesIncluded <String[]>] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description
Create a new RemotePCAccount. A RemotePCAccount defines machine filters to support Remote PC automation adding unconfigured machines to catalogs.

Related topics
Get-BrokerRemotePCAccount
Set-BrokerRemotePCAccount
Remove-BrokerRemotePCAccount

Parameters
-CatalogUid <Int32>
Specifies the catalog which Remote PC automation adds an unconfigured machine to if it matches this RemotePCAccount.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-OU <String>
Specifies the DN of an AD container, or has the special value ‘any’.

When an AD container is specified a machine may only match with the RemotePCAccount when the AD computer object is located relative to the OU.

When ‘any’ is specified the location of the AD computer object is ignored for purposes of matching this RemotePCAccount. The machine must still meet the MachinesIncluded and MachinesExcluded filters for a match to occur.

In the event that a machine matches with multiple RemotePCAccounts then the RemotePCAccount OU with the longest canonical name takes precedence. The special ‘any’ OU is treated as lowest priority.

Note that the OU value of every RemotePCAccount must be unique, and this includes only one ‘any’ entry being permitted.

| Required? | true |
| Default Value | null |
| Accept Pipeline Input? | true (ByPropertyName) |

-AllowSubfolderMatches <Boolean>

When true a machine matches this RemotePCAccount if the AD computer object exists within the container specified by the OU property, or within a child container of the OU.

When false the AD computer object only matches if it exists directly in the AD container specified by the OU property.
This property is not meaningful when OU has the special value ‘any’.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-MachinesExcluded<String[]>

MachinesExcluded specifies a set of strings that can include asterisk wildcards. If a machine name matches any entries in MachinesExcluded then it cannot match with this RemotePCAccount regardless of whether there is a MachinesIncluded match.

Matches are performed against the domain name joined with the machine name by a backslash (DOMAIN\MACHINE), e.g.:

- DOMAIN1\M*
- DOMAIN*\M*
- *\M*

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>@()</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-MachinesIncluded<String[]>

MachinesIncluded specifies a set of strings that can include asterisk wildcards. A machine may only match with this RemotePCAccount if it matches a MachinesIncluded entry and does not match any MachinesExcluded entries.

Matches are performed against the domain name joined with the machine name by a backslash (DOMAIN\MACHINE), e.g.:

- DOMAIN1\M*
- DOMAIN*\M*
- *\M*

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>@('*')</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-LoggingId<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.
## New-BrokerRemotePCAccount

**Required?** false

**Default Value** Localhost. Once a value is provided by any cmdlet, this value will become the default.

**Accept Pipeline Input?** false

### Input Type

None You cannot pipe input into this cmdlet.

### Return Values

Citrix.BrokerAdmin.SDK.RemotePCAccount

The newly created RemotePCAccount.

### Examples

------------ EXAMPLE 1 ---------------

C:\PS> New-BrokerRemotePCAccount -OU 'ou=MyOU,dc=MyDomain,dc=com' -CatalogUid 42
Create a RemotePCAccount that adds unconfigured machines with computer objects in MyOU, into catalog 42.

------------ EXAMPLE 2 ---------------

C:\PS> New-BrokerRemotePCAccount -OU 'any' -CatalogUid 42 -MachinesIncluded @('DOMAIN1\*') -MachinesExcluded @('DOMAIN1\JOHNOE*')
Create a RemotePCAccount matching unconfigured machines from DOMAIN1, except those with hostnames containing JOHND0E, and add them to catalog 42.

------------ EXAMPLE 3 ---------------

C:\PS> New-BrokerRemotePCAccount -OU 'any' -CatalogUid 42
Create a RemotePCAccount that matches any unconfigured machine, causing automation to add matching machines to catalog 42.
New-BrokerTag

Oct 10, 2013
Creates a new tag.

Syntax

New-BrokerTag [-Name <String> [-UUID <Guid>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Creates a tag that can be associated with other objects using Add-BrokerTag.

Related topics

Add-BrokerTag
Get-BrokerTag
Remove-BrokerTag
Rename-BrokerTag

Parameters

-**Name**<String>

Specifies a name for the new tag.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**UUID**<Guid>

Specifies a UUID for the new tag. When not specified, a UUID is automatically assigned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.
- **AdminAddress** `<String>`

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a hostname or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

None Input cannot be piped to this cmdlet.

**Return Values**

`Citrix.BrokerAdmin.SDK.Tag`

Outputs the generated tag.

**Examples**

```
-------------------------- EXAMPLE 1 --------------------------
C:\PS> New-BrokerTag -Name 'Tag1'
Creates a new tag with name 'Tag1'.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> New-BrokerTag 'Tag2' | Add-BrokerTag -Desktop $desktop
Creates a new tag with name 'Tag2' and associates it with Desktop $desktop.
```
New-BrokerUser

Oct 10, 2013
Creates a new broker user object

Syntax

New-BrokerUser [-SID] <SecurityIdentifier> [-AdminAddress <String>] [<CommonParameters>]
New-BrokerUser [-Name] <String> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The New-BrokerUser cmdlet creates a new broker object to represent a user identity (or the identity of a group of users). The object is created local to the PowerShell environment in which the cmdlet is run; no new user object is created in the broker configuration, unless the object is added to another broker object, such as a machine or a desktop. For details, see Add-BrokerUser.

The identity of the user or group must be specified using either the Name or SID parameter

Related topics

Add-BrokerUser
Get-BrokerUser
Remove-BrokerUser

Parameters

-SID <SecurityIdentifier>
The SID of the user or group

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>null</td>
<td>false</td>
</tr>
</tbody>
</table>

-Name <String>
The name of the user or group

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>null</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None

You cannot pipe input into this cmdlet.

Return Values

Citrix.BrokerAdmin.SDK.User

The broker user object

Notes

Typically, broker user objects are created implicitly using the Add-BrokerUser cmdlet with a user name or SID.

Examples

-------------------------- EXAMPLE 1 --------------------------

$user = New-BrokerUser DOMAIN\UserName
Create a broker user object for the specified user.

-------------------------- EXAMPLE 2 --------------------------

$user = New-BrokerUser -SID S-1-5-23-1763203430-193137401-908696819-3450
Create a broker user object for the specified user.
Remove-BrokerAccessPolicyRule

Oct 10, 2013
Deletes a rule from the site's access policy.

Syntax


Remove-BrokerAccessPolicyRule [-Name] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Remove-BrokerAccessPolicyRule cmdlet deletes a rule from the site's access policy.

An access policy rule defines a set of connection filters and access control rights relating to a desktop group. These allow fine-grained control of what access is granted to a desktop group based on details of, for example, a user's endpoint device, its address, and the user's identity.

Deleting a rule does not affect existing user sessions, but it may result in users being unable to launch new sessions, or reconnect to disconnected sessions if access to the desktop group delivering those sessions was granted by the deleted rule.

Related topics

New-BrokerAccessPolicyRule
Get-BrokerAccessPolicyRule
Set-BrokerAccessPolicyRule
Rename-BrokerAccessPolicyRule

Parameters

-**InputObject** <AccessPolicyRule[]>
The access policy rule to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Name** <String>
The name of the access policy rule to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
**Default Value**

| Accept Pipeline Input? | true (ByPropertyName) |

**-LoggingId<Guid>**

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**

Citrix.BrokerAdmin.SDK.AccessPolicyRule The access policy rule to be deleted.

**Return Values**

None

**Examples**

------------ EXAMPLE 1 ---------------

C:\PS> Remove-BrokerAccessPolicyRule 'Temp Staff'
Deletes the access policy rule called Temp Staff. Existing sessions are not affected, but if access was granted by the deleted rule users may be unable to reconnect to sessions if they are subsequently disconnected.

------------ EXAMPLE 2 ---------------

Deletes all access policy rules explicitly granting user SALES\johndoe access to any desktop group in the site. Any existing desktop sessions for the user are not affected. The user may still be able to access site resources by access policy rules that grant access through group membership or non-user-based connection filters.
Remove-BrokerAccessPolicyRuleMetadata

Oct 10, 2013

Syntax


Detailed Description


Related topics

Parameters

- **-InputObject** <AccessPolicyRule[]>

  Specifies the AccessPolicyRule object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>

  Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type


Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerAccessPolicyRuleMetadata -InputObject $obj-Uid -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for the AccessPolicyRule whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerAccessPolicyRuleMetadata -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for all the AccessPolicyRule in the site
Remove-BrokerAppAssignmentPolicyRule

Oct 10, 2013
Deletes an application rule from the site’s assignment policy.

Syntax


Remove-BrokerAppAssignmentPolicyRule [-Name] <String> [-LoggingId <Guid>] [-AdminAddress "String"] [<CommonParameters>]

Detailed Description

The Remove-BrokerAppAssignmentPolicyRule cmdlet deletes an application rule from the site’s assignment policy.

An application rule in the assignment policy defines the users who are entitled to a self-service persistent machine assignment from the rule’s desktop group; once assigned the machine can run one or more applications published from the group.

Deleting an application rule does not remove machine assignments that have already been made by the rule, nor does it affect active sessions to those machines in any way.

Related topics

New-BrokerAppAssignmentPolicyRule
Get-BrokerAppAssignmentPolicyRule
Set-BrokerAppAssignmentPolicyRule
Rename-BrokerAppAssignmentPolicyRule

Parameters

-InputObject <AppAssignmentPolicyRule[]>-

The application rule to be deleted from the assignment policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name <String>

The name of the application rule to be deleted from the assignment policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1190
## Default Value

| Accept Pipeline Input? | true (ByPropertyName) |

### -LoggingId <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

### Input Type

Citrix.BrokerAdmin.SDK.AppAssignmentPolicyRule The application rule to be deleted from the assignment policy.

### Return Values

None

### Examples

```
-------------------------- EXAMPLE 1 --------------------------
C:\PS> Remove-BrokerAppAssignmentPolicyRule 'Temp Staff'
Deletes the application rule called Temp Staff from the assignment policy. Access to machines already assigned by this rule is not affected in any way.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> $dg = Get-BrokerDesktopGroup 'Sales Support'
Deletes the application rule for the Sales Support desktop group from the site's assignment policy. This prevents any further machine assignments being made from this group, but it does not affect existing assignments made by the rule.
```
Remove-BrokerAppEntitlementPolicyRule

Oct 10, 2013
Deletes an application rule from the site's entitlement policy.

Syntax


Remove-BrokerAppEntitlementPolicyRule [-Name] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Remove-BrokerAppEntitlementPolicyRule cmdlet deletes an application rule from the site's entitlement policy.

An application rule in the entitlement policy defines the users who are allowed per-session access to a machine to run one or more applications published from the rule's desktop group.

Deleting a rule does not affect existing sessions launched using the rule, but users cannot reconnect to those sessions if they are subsequently disconnected.

Related topics

New-BrokerAppEntitlementPolicyRule
Get-BrokerAppEntitlementPolicyRule
Set-BrokerAppEntitlementPolicyRule
Rename-BrokerAppEntitlementPolicyRule

Parameters

- **-InputObject** <AppEntitlementPolicyRule[]>

The application rule to be deleted from the entitlement policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>

The name of the application rule to be deleted from the entitlement policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | true (ByPropertyName)
---|---

**-LoggingId (Guid)**

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress (String)**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdminSDK.AppEntitlementPolicyRule The application rule to be deleted from the entitlement policy.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerAppEntitlementPolicyRule 'Temp Workers'

Deletes the application rule called Temp Workers from the entitlement policy rule. Existing application sessions launched using that rule are not affected, but users cannot reconnect to those sessions if they are subsequently disconnected.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> $dg = Get-BrokerDesktopGroup 'Customer Support'

Deletes the application rule from the entitlement policy rule applied to the Customer Support desktop group. This effectively removes all access to the applications published from this group. Existing application sessions are not affected, but users cannot reconnect to those sessions if they are subsequently disconnected.
Remove-BrokerApplication

Oct 10, 2013
Deletes one or more applications, or an association of an application.

Syntax


Detailed Description

The Remove-BrokerApplication cmdlet deletes one or more applications, or you can use it to delete just the association of an application to a desktop group.

Its usage dictates the behavior of the cmdlet. If only the application is specified as a parameter, then the cmdlet deletes the application. It also deletes any associations this application has with other objects, such as with access policy rules or desktop groups. More specifically, when an application is deleted the following happens:

- The association to any desktop groups is removed.
- The association to any tags is removed.
- Any configured file-type association objects for this application are deleted.
- The association to any user accounts is removed.
- The association to any access session conditions is removed.
- The access policy rule object for this application, if one existed, is deleted.
- Finally, the application object itself is deleted.

Note that if the application is in use by a user then the application cannot be deleted.

If more than just the application is supplied as a parameter to the cmdlet (for instance, if a DesktopGroup object is also specified) then the application is not deleted. Instead, only the association from the application to that desktop group is removed.

Related topics

New-BrokerApplication
Add-BrokerApplication
Get-BrokerApplication
Rename-BrokerApplication
Set-BrokerApplication

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
Parameters

- **InputObject** `<Application[]>`

  Specifies the applications to delete. TheUid can also be substituted for the application objects.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name** `<String>`

  Specifies the name of the application to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByPropertyName)</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **Force** `<SwitchParameter>`

  Remove application even if it's in use. Removing an application that is currently in use, can potentially leave an application session containing no applications. If all the applications that are currently active in a disconnected application session are removed, the user will be unable to reconnect to the session. Forcing removal of an in-use application does not impact the actual session itself.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **DesktopGroup** `<DesktopGroup>`

  Specifies the desktop group that this application should no longer be associated with. The Uid or Name can also be substituted for the desktop group object.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>true (ByValue)</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **LoggingId** `<Guid>`

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level
Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**:<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

`Citrix.BrokerAdmin.SDK.Application` The application objects can be specified as input.

**Return Values**

None

This cmdlet does not return any output.

**Examples**

---------- EXAMPLE 1 ----------

C:\PS> Remove-BrokerApplication "Notepad"
This command deletes the application that has a BrowserName of "Notepad".

---------- EXAMPLE 2 ----------

C:\PS> $app = Get-BrokerApplication -BrowserName "Notepad"
C:\PS> $group = Get-BrokerDesktopGroup -Name "Private DesktopGroup"
C:\PS> Remove-BrokerApplication -InputObject $app -DesktopGroup $group
This command removes the association of the desktop group that has a name of "Private DesktopGroup" from the application that has a BrowserName of "Notepad". It does not otherwise modify the application.
Remove-BrokerApplicationInstanceMetadata

Oct 10, 2013
Deletes ApplicationInstance Metadata from the ApplicationInstance objects

Syntax

Remove-BrokerApplicationInstanceMetadata [-InputObject] <ApplicationInstance[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Remove-BrokerApplicationInstanceMetadata cmdlet deletes Metadata from the ApplicationInstance objects.

Related topics

Parameters

- **-InputObject**<ApplicationInstance[]>

  Specifies the ApplicationInstance object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name**<String>

  Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId**<Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerApplicationInstance You can pipe the ApplicationInstance to delete the metadata.

**Return Values**

None

**Examples**

-------------- EXAMPLE 1 ---------------

C:\PS> Remove-BrokerApplicationInstanceMetadata -InputObject $obj-Uid -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for the ApplicationInstance whose instance is pointed by $obj-Uid

-------------- EXAMPLE 2 ---------------

C:\PS> Remove-BrokerApplicationInstanceMetadata -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for all the ApplicationInstance in the site
Remove-BrokerApplicationMetadata

Oct 10, 2013
Deletes Application Metadata from the Application objects

Syntax

Remove-BrokerApplicationMetadata [-InputObject] <Application[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Remove-BrokerApplicationMetadata cmdlet deletes Metadata from the Application objects.

Related topics

Parameters

- **-InputObject** <Application[]>
  
  Specifies the Application object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>
  
  Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>
  
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**: `<String>`

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerApplication You can pipe the Application to delete the metadata.

**Return Values**

None

**Examples**

------------------------------- EXAMPLE 1 -------------------------------

C:\PS> Remove-BrokerApplicationMetadata -InputObject $obj-Uid -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for the Application whose instance is pointed by $obj-Uid

------------------------------- EXAMPLE 2 -------------------------------

C:\PS> Remove-BrokerApplicationMetadata -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for all the Application in the site
Remove-BrokerAssignmentPolicyRule

Oct 10, 2013
Deletes a desktop rule from the site's assignment policy.

Syntax


Remove-BrokerAssignmentPolicyRule [-Name] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<<CommonParameters>>]

Detailed Description

The Remove-BrokerAssignmentPolicyRule cmdlet deletes a desktop rule from the site's assignment policy.

A desktop rule in the assignment policy defines the users who are entitled to self-service persistent machine assignments from the rule's desktop group. A rule defines how many machines a user is allowed from the group for delivery of full desktop sessions.

Deleting a desktop rule does not remove machine assignments that have already been made by the rule, nor does it affect active sessions to those machines in any way.

Related topics

New-BrokerAssignmentPolicyRule

Get-BrokerAssignmentPolicyRule

Set-BrokerAssignmentPolicyRule

Rename-BrokerAssignmentPolicyRule

Parameters

-InputObject <AssignmentPolicyRule[]>

The desktop rule to be deleted from the assignment policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name <String>

The name of the desktop rule to be deleted from the assignment policy.

<table>
<thead>
<tr>
<th>Required?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
</tr>
</tbody>
</table>
**Default Value**

| Accept Pipeline Input? | true (ByPropertyName) |

- **LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**

Citrix.BrokerAdmin.SDK.AssignmentPolicyRule The desktop rule to be deleted from the assignment policy.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerAssignmentPolicyRule 'Temp Staff'

Deletes the desktop rule called Temp Staff from the assignment policy. Access to machines already assigned by this rule is not affected in any way.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> $dg = Get-BrokerDesktopGroup 'Sales Support'
C:\PS> Get-BrokerAssignmentPolicyRule -DesktopGroupId $dg.Uid | Remove-BrokerAssignmentPolicyRule

Deletes all desktop rules for the Sales Support desktop group from the site's assignment policy. This prevents any further machine assignments being made from this group, but it does not affect existing assignments made by these rules.
Remove-BrokerAssignmentPolicyRuleMetadata

Oct 10, 2013
Deletes AssignmentPolicyRule Metadata from the AssignmentPolicyRule objects

Syntax


Detailed Description


Related topics

Parameters

- **InputObject** <AssignmentPolicyRule[]>

Specifies the AssignmentPolicyRule object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name** <String>

Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId** <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**: `<String>`

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

  **Input Type**

  `Citrix.BrokerAdmin.SDK.BrokerAssignmentPolicyRule` You can pipe the AssignmentPolicyRule to delete the metadata.

  **Return Values**

  None

  **Examples**

  --------------- EXAMPLE 1 ---------------

  ```
  C:\PS> Remove-BrokerAssignmentPolicyRuleMetadata -InputObject $obj-Uid -Name "MyMetadataName"
  ```

  This command deletes the Metadata "MyMetadataName" key-value pair for the AssignmentPolicyRule whose instance is pointed by $obj-Uid

  --------------- EXAMPLE 2 ---------------

  ```
  C:\PS> Remove-BrokerAssignmentPolicyRuleMetadata -Name "MyMetadataName"
  ```

  This command deletes the Metadata "MyMetadataName" key-value pair for all the AssignmentPolicyRule in the site
Remove-BrokerCatalog

Oct 10, 2013
Removes catalogs from the site.

Syntax

Remove-BrokerCatalog [-InputObject] <Catalog[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]
Remove-BrokerCatalog [-Name] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Remove catalogs from the site.

In order to remove a catalog from a site, the catalog must not contain machines. To remove a machine from a catalog use the Remove-BrokerMachine cmdlet. Note: in order to remove a machine from a catalog, it must not belong to a desktop group.

Related topics

Add-BrokerCatalog
Get-BrokerCatalog
New-BrokerCatalog
Rename-BrokerCatalog
Set-BrokerCatalog
Add-BrokerDesktopGroup
Remove-BrokerDesktopGroup

Parameters

- **InputObject**<Catalog[]>

Specifies the catalog objects to delete.

<table>
<thead>
<tr>
<th>Required?</th>
<th>True</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name**<String>

Specifies the name of the catalog to delete.

<table>
<thead>
<tr>
<th>Required?</th>
<th>True</th>
</tr>
</thead>
</table>
Default Value

Accept Pipeline Input?

true (ByPropertyName)

-LoggingId<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Required? false

Default Value

Accept Pipeline Input? false

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

Required? false

Default Value

Localhost. Once a value is provided by any cmdlet, this value will become the default.

Accept Pipeline Input? false

Input Type

Citrix.BrokerAdmin.SDK.Catalog You can pipe the catalogs to be deleted to Remove-BrokerCatalog.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerCatalog -Name "MyCatalog"
C:\PS> Remove-BrokerCatalog -InputObject (Get-BrokerCatalog -Name "MyCatalog")
These commands delete the catalog with the name "MyCatalog".

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerCatalog -Name 'test*'  
This command deletes all catalogs with names beginning with "test".

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Get-BrokerCatalog -RemotePCDesktopGroupUid 42 | Remove-BrokerCatalog -RemotePCDesktopGroup 42  
Remove all the Remote PC catalogs that are associated with desktop group 42. Note that this only breaks the Remote PC
relationships and does not delete the desktop groups.
Remove-BrokerCatalogMetadata

Oct 10, 2013
Deletes Catalog Metadata from the Catalog objects

Syntax

Remove-BrokerCatalogMetadata [-InputObject] <Catalog[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Remove-BrokerCatalogMetadata cmdlet deletes Metadata from the Catalog objects.

Related topics

Parameters

- **-InputObject**<Catalog[]>

  Specifies the Catalog object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

  | Accept Pipeline Input? | true (ByValue) |

- **-Name**<String>

  Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (false)</td>
</tr>
</tbody>
</table>

- **-LoggingId**<Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerCatalog You can pipe the Catalog to delete the metadata.

**Return Values**

None

**Examples**

------------ EXAMPLE 1 ---------------

C:\PS> Remove-BrokerCatalogMetadata -InputObject $obj-Uid -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for the Catalog whose instance is pointed by $obj-Uid

------------ EXAMPLE 2 ---------------

C:\PS> Remove-BrokerCatalogMetadata -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for all the Catalog in the site
Remove-BrokerConfigurationSlot

Oct 10, 2013
Removes a configuration slot.

Syntax

Remove-BrokerConfigurationSlot [-InputObject] <ConfigurationSlot[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-BrokerConfigurationSlot [-Name] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Removes a configuration slot from the site. All machine configurations associated with this slot are also removed.

Related topics

New-BrokerConfigurationSlot
Get-BrokerConfigurationSlot

Parameters

-**InputObject** <ConfigurationSlot[]>

Configuration slot to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Name** <String>

Name of configuration slot to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**LoggingId** <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Configuration slot to remove Configuration slots may be specified through pipeline input.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

Remove-BrokerConfigurationSlot -Name "User Profile Manager"

Remove the configuration slot named "User Profile Manager".
Remove-BrokerConfigurationSlotMetadata

Oct 10, 2013
Deletes ConfigurationSlot Metadata from the ConfigurationSlot objects

Syntax

Remove-BrokerConfigurationSlotMetadata [-InputObject] <ConfigurationSlot[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Remove-BrokerConfigurationSlotMetadata cmdlet deletes Metadata from the ConfigurationSlot objects.

Related topics

Parameters

- **-InputObject** <ConfigurationSlot[]>

Specifies the ConfigurationSlot object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>

Specifies the name of the Metadata to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
-AdminAddress\<String>\n
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdmin.SDK.BrokerConfigurationSlot You can pipe the ConfigurationSlot to delete the metadata.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerConfigurationSlotMetadata -InputObject $obj-Uid -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for the ConfigurationSlot whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerConfigurationSlotMetadata -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for all the ConfigurationSlot in the site
Remove-BrokerConfiguredFTA

Oct 10, 2013
Deletes one or more configured file type associations.

Syntax


Detailed Description

Deletes one or more file type associations configured for a published application. At least one configured file type association object must be specified.

Related topics

Get-BrokerConfiguredFTA

New-BrokerConfiguredFTA

Parameters

- **InputObject**<ConfiguredFTA[]>
  
  Specifies the ConfiguredFTA objects to delete.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>
  
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>
  
  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdmin.SDK.ConfiguredFTA[] One or more ConfiguredFTA objects can be supplied as input.

Return Values

None

Examples

```
-------------------------- EXAMPLE 1 --------------------------
C:\PS> $ftas = Get-BrokerConfiguredFTA -ExtensionName ".txt"
C:\PS> Remove-BrokerConfiguredFTA $ftas
Deletes all configured file type associations with an extension name of ".txt".
```
Remove-BrokerControllerMetadata

Oct 10, 2013
Deletes Controller Metadata from the Controller objects

Syntax

Remove-BrokerControllerMetadata [-InputObject] <Controller[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Remove-BrokerControllerMetadata cmdlet deletes Metadata from the Controller objects.

Related topics

Parameters

- **InputObject** <Controller[]>
  Specifies the Controller object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name** <String>
  Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

- **LoggingId** <Guid>
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**<String>

 Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerController You can pipe the Controller to delete the metadata.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerControllerMetadata -InputObject $obj-Uid -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for the Controller whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerControllerMetadata -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for all the Controller in the site
Remove-BrokerDelayedHostingPowerAction

Oct 10, 2013
Cancels one or more delayed power actions.

Syntax

Remove-BrokerDelayedHostingPowerAction [-InputObject] <DelayedHostingPowerAction[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-BrokerDelayedHostingPowerAction [-MachineName] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Removes one or more delayed power actions that have not yet been queued for execution.

Related topics
Get-BrokerDelayedHostingPowerAction
New-BrokerDelayedHostingPowerAction

Parameters

-**InputObject** `<DelayedHostingPowerAction[]>`

The power action to be cancelled.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**MachineName** `<String>`

Cancels only actions for machines whose name (of the form domain\machine) matches the specified string.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-**LoggingId** `<Guid>`

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.
Required? | false
---|---
Default Value | 
Accept Pipeline Input? | false

-AdminAddress

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

```
Required? | false
---|---
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
Accept Pipeline Input? | false
```

Input Type

Citrix.BrokerAdmin.SDK.DelayedHostingPowerAction The power action to be cancelled.

Return Values

None

Examples

```
-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerHostingPowerAction -MachineName 'XD_VDA1'
Cancels any pending delayed power actions for the machine called XD_VDA1.
```

Remove-BrokerDesktopGroup

Oct 10, 2013
Remove desktop groups from the system or remove them from a Remote PC catalog.

Syntax


Remove-BrokerDesktopGroup [-Name] <String> [-Force] [-RemotePCCatalog <Catalog>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

This cmdlet has 2 functions:

o Remove desktop groups from the system.

o Break Remote PC associations between desktop groups and a catalog.

The Remote PC relationships are used by Remote PC automation to determine which desktop groups a machine in a particular Remote PC catalog can be published to. The assignment policy rules belonging to those desktop groups also determines the set of users that are allowed to be assigned to machines from the catalog.

Related topics

Get-BrokerDesktopGroup
New-BrokerDesktopGroup
Set-BrokerDesktopGroup
Add-BrokerDesktopGroup
Rename-BrokerDesktopGroup
Add-BrokerCatalog
Remove-BrokerCatalog

Parameters

- **InputObject** <DesktopGroup[]>

  Specifies the desktop groups to remove.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>InputObject</strong></td>
<td>true</td>
<td>null</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>


-Name<String>

Specifies the name of the desktop group to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Force<SwitchParameter>

Remove desktop groups even if there are active sessions.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-RemotePCCatalog<Catalog>

When this parameter is specified, Remote PC desktop groups are removed from the specified Remote PC catalog.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-LoggingId<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | false

Input Type
Citrix.BrokerAdmin.SDK.DesktopGroup You can pipe desktop groups to Remove-BrokerDesktopGroup.

Return Values
None

Notes
If a desktop group contains desktops when it is removed, these desktops are also removed (but the underlying broker machine remains).

A desktop group that still has active sessions cannot be removed unless the -Force switch is used.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerDesktopGroup EMEA*
Remove all desktop groups with names starting with "EMEA".

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerDesktopGroup -Enabled $false | Remove-BrokerDesktopGroup -Force
Remove all desktops that are currently disabled even if there are active sessions.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Get-BrokerDesktopGroup -RemotePCCatalogUid 42 | Remove-BrokerDesktopGroup -RemotePCCatalog 42
Remove all the Remote PC desktop groups that are associated with catalog 42. Note that this only breaks the Remote PC relationships and does not delete the desktop groups.
Remove-BrokerDesktopGroupMetadata

Oct 10, 2013
Deletes DesktopGroup Metadata from the DesktopGroup objects

Syntax

Remove-BrokerDesktopGroupMetadata [-InputObject] <DesktopGroup[]> [-Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Remove-BrokerDesktopGroupMetadata cmdlet deletes Metadata from the DesktopGroup objects.

Related topics

Parameters

- **InputObject** <DesktopGroup[]>

  Specifies the DesktopGroup object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name** <String>

  Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId** <Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
**AdminAddress**:<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerDesktopGroup You can pipe the DesktopGroup to delete the metadata.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerDesktopGroupMetadata -InputObject $obj-Uid -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for the DesktopGroup whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerDesktopGroupMetadata -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for all the DesktopGroup in the site
Remove-BrokerEntitlementPolicyRule

Oct 10, 2013
Deletes a desktop rule from the site's entitlement policy.

Syntax


Remove-BrokerEntitlementPolicyRule [-Name] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Remove-BrokerEntitlementPolicyRule cmdlet deletes a desktop rule from the site's entitlement policy.

A desktop rule in the entitlement policy defines the users who are allowed per-session access to a machine from the rule’s associated desktop group to run a full desktop session.

Deleting a rule does not affect existing sessions launched using the rule, but users cannot reconnect to those sessions if they are subsequently disconnected.

Related topics

New-BrokerEntitlementPolicyRule
Get-BrokerEntitlementPolicyRule
Set-BrokerEntitlementPolicyRule
Rename-BrokerEntitlementPolicyRule

Parameters

- **InputObject** <EntitlementPolicyRule[]>

  The desktop rule to be deleted from the entitlement policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

- **Name** <String>

  The name of the desktop rule to be deleted from the entitlement policy.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? true (ByProperty Name)

-LoggingId<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdmin.SDK.EntitlementPolicyRule The desktop rule to be deleted from the entitlement policy.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Remove-BrokerEntitlementPolicyRule 'Temp Workers'
Deletes the desktop rule called Temp Workers from the entitlement policy. Existing desktop sessions launched using the rule are not affected, but users cannot reconnect to sessions if they are subsequently disconnected.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> $dg = Get-BrokerDesktopGroup 'Customer Support'
Deletes all desktop rules from the entitlement policy applying to the Customer Support desktop group. This effectively removes all access to the desktops published from this group. Existing desktop sessions are not affected, but users cannot reconnect to sessions if they are subsequently disconnected.
Remove-BrokerEntitlementPolicyRuleMetadata

Oct 10, 2013
Deletes EntitlementPolicyRule Metadata from the EntitlementPolicyRule objects

Syntax


Detailed Description


Related topics

Parameters

- **-InputObject** <EntitlementPolicyRule[]>
  Specifies the EntitlementPolicyRule object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>
  Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**: String

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

  | Required? | false |
  | Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
  | Accept Pipeline Input? | false |

  **Input Type**


  **Return Values**

  None

  **Examples**

  -------------------------- EXAMPLE 1 --------------------------

  C:\PS> Remove-BrokerEntitlementPolicyRuleMetadata -InputObject $obj-Uid -Name "MyMetadataName"

  This command deletes the Metadata "MyMetadataName" key-value pair for the EntitlementPolicyRule whose instance is pointed by $obj-Uid

  -------------------------- EXAMPLE 2 --------------------------

  C:\PS> Remove-BrokerEntitlementPolicyRuleMetadata -Name "MyMetadataName"

  This command deletes the Metadata "MyMetadataName" key-value pair for all the EntitlementPolicyRule in the site
Remove-BrokerHostingPowerAction

Oct 10, 2013
Cancel one or more pending power actions.

Syntax

Remove-BrokerHostingPowerAction [-InputObject] <HostingPowerAction[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-BrokerHostingPowerAction [-MachineName] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Causes one or more of the pending power actions in the queue to be marked as cancelled. The affected power actions are not sent to the hypervisor for processing, and take no further part in the queuing activity.

Power actions cannot be cancelled once they have started to be processed by the hypervisor.

Related topics

Get-BrokerHostingPowerAction

New-BrokerHostingPowerAction

Set-BrokerHostingPowerAction

Parameters

- **-InputObject** <HostingPowerAction[]>

  The power action to be cancelled.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-MachineName** <String>

  Cancels only actions for machines whose name (of the form domain\machine) matches the specified string.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.HostingPowerAction The power action to be cancelled.

**Return Values**

None

**Examples**

------------- EXAMPLE 1 ---------------

C:\PS> Remove-BrokerHostingPowerAction -MachineName 'XD_VDA1'
Cancels any pending power actions for the machine called 'XD_VDA1'.

------------- EXAMPLE 2 ---------------

C:\PS> Get-BrokerHostingPowerAction -Filter { State -eq "Pending" -and RequestTime -gt "-00:05" } | Remove-BrokerHostingPowerAction
Cancels any pending power actions requested in the last five minutes.
Remove-BrokerHostingPowerActionMetadata

Oct 10, 2013
Deletes HostingPowerAction Metadata from the HostingPowerAction objects

Syntax

Remove-BrokerHostingPowerActionMetadata [-InputObject] <HostingPowerAction[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Remove-BrokerHostingPowerActionMetadata cmdlet deletes Metadata from the HostingPowerAction objects.

Related topics

Parameters

- **-InputObject** <HostingPowerAction[]>
  Specifies the HostingPowerAction object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>
  Specifies the name of the Metadata to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**: <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerHostingPowerAction You can pipe the HostingPowerAction to delete the metadata.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerHostingPowerActionMetadata -InputObject $obj-Uid -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for the HostingPowerAction whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerHostingPowerActionMetadata -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for all the HostingPowerAction in the site
Remove-BrokerHypervisorAlertMetadata

October 10, 2013
Deletes HypervisorAlert Metadata from the HypervisorAlert objects

Syntax

Remove-BrokerHypervisorAlertMetadata [-InputObject] <HypervisorAlert[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters>

Detailed Description

The Remove-BrokerHypervisorAlertMetadata cmdlet deletes Metadata from the HypervisorAlert objects.

Related topics

Parameters

- **InputObject** <HypervisorAlert[]>

  Specifies the HypervisorAlert object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name**<String>

  Specifies the name of the Metadata to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
**-AdminAddress**: <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerHypervisorAlert You can pipe the HypervisorAlert to delete the metadata.

**Return Values**

None

**Examples**

------------ EXAMPLE 1 ---------------

C:\PS> Remove-BrokerHypervisorAlertMetadata -InputObject $obj-Uid -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for the HypervisorAlert whose instance is pointed by $obj-Uid

------------ EXAMPLE 2 ---------------

C:\PS> Remove-BrokerHypervisorAlertMetadata -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for all the HypervisorAlert in the site
Remove-BrokerHypervisorConnection

Oct 10, 2013
Removes a hypervisor connection from the system.

Syntax

Remove-BrokerHypervisorConnection [-InputObject] <HypervisorConnection[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-BrokerHypervisorConnection [-Name] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Remove-BrokerHypervisorConnection removes a hypervisor connection from the system. A hypervisor connection cannot be removed if it's being used by a machine.

Related topics

Get-BrokerHypervisorConnection
Set-BrokerHypervisorConnection
New-BrokerHypervisorConnection

Parameters

-**InputObject** <HypervisorConnection[]>

Specifies the hypervisor connection object to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Name** <String>

Specifies the name of the hypervisor connection object to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**LoggingId** <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop
Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**

Citrix.BrokerAdmin.SDK.HypervisorConnection You can pipe the hypervisor connection to be removed to Remove-BrokerHypervisorConnection.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

c:\PS> Remove-BrokerHypervisorConnection -Name "Xen Server Connection"
This command removes a hypervisor connection by name.

-------------------------- EXAMPLE 2 --------------------------

c:\PS> $hvConn = Get-BrokerHypervisorConnection -PreferredController "controllerName" -Name "Xen Server Connection"
c:\PS> Remove-BrokerHypervisorConnection -InputObject $hvConn
Gets a hypervisor connection by preferred controller and removes it.
Remove-BrokerHypervisorConnectionMetadata

Oct 10, 2013
Deletes HypervisorConnection Metadata from the HypervisorConnection objects

Syntax

```
Remove-BrokerHypervisorConnectionMetadata [-InputObject] <HypervisorConnection[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
```

Detailed Description

The Remove-BrokerHypervisorConnectionMetadata cmdlet deletes Metadata from the HypervisorConnection objects.

Related topics

Parameters

- **-InputObject** `<HypervisorConnection[]>`

  Specifies the HypervisorConnection object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** `<String>`

  Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true</td>
</tr>
</tbody>
</table>

- **-LoggingId** `<Guid>`

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
**AdminAddress**: `<String>`

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

`Citrix.BrokerAdmin.SDK.BrokerHypervisorConnection` You can pipe the `HypervisorConnection` to delete the metadata.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

```powershell
C:\PS> Remove-BrokerHypervisorConnectionMetadata -InputObject $obj-Uid -Name "MyMetadataName"
```

This command deletes the Metadata "MyMetadataName" key-value pair for the HypervisorConnection whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

```powershell
C:\PS> Remove-BrokerHypervisorConnectionMetadata -Name "MyMetadataName"
```

This command deletes the Metadata "MyMetadataName" key-value pair for all the HypervisorConnection in the site
Remove-BrokerIcon

Oct 10, 2013
Remove an icon.

Syntax

Remove-BrokerIcon [[-InputObject] <Icon[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]]

Detailed Description

Removes an icon from the database.

Related topics
Add-BrokerIcon
New-BrokerIcon

Parameters

-**InputObject** <Icon[]>
  
  Specifies the icon to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | false

Input Type
Citrix.BrokerAdmin.SDK.Icon The icon to be removed can be piped into the cmdlet.

Return Values
None

Notes
Note that if the icon is currently in use, for example, as a desktop icon, it cannot be removed until the association is cleared.

Examples
-------------------------- EXAMPLE 1 --------------------------
C:\PS> Remove-BrokerIcon 3
Removes the icon with Uid 3.
Remove-BrokerIconMetadata

Oct 10, 2013
Deletes Icon Metadata from the Icon objects

Syntax

Remove-BrokerIconMetadata [-InputObject] <Icon[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
The Remove-BrokerIconMetadata cmdlet deletes Metadata from the Icon objects.

Related topics

Parameters

- **-InputObject** <Icon[]>
  Specifies the Icon object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>
  Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
-AdminAddress\<String>\>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdmin.SDK.BrokerIcon You can pipe the Icon to delete the metadata.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerIconMetadata -InputObject $obj-Uid -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for the Icon whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerIconMetadata -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for all the Icon in the site
Remove-BrokerImportedFTA

Oct 10, 2013
Deletes one or more imported file type associations.

Syntax

Remove-BrokerImportedFTA -DesktopGroupUids <Int32[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Deletes all of the imported file type associations belonging to one or more desktop groups. At least one desktop group must be specified.

Imported file type associations are grouped together based on the desktop group of the machine from which they were imported. All file types for a desktop group are deleted. There is no mechanism for deleting a subset imported file type associations for a specific desktop group.

Imported file type associations are different from configured file type associations. Imported file type associations are lists of known file type associations for a given desktop group. Configured file type associations are those that are actually associated with published applications for the purposes of content redirection.

Related topics

Get-BrokerImportedFTA
Update-BrokerImportedFTA

Parameters

-**DesktopGroupUids**<Int32[]>

Deletes the imported file type associations belonging to specified desktop groups.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | false  
---|---

**-AdminAddress $String**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Int32[] An array of Uids for the desktop groups can be supplied as input. The desktop groups must be of the Private or Shared desktop kind.

**Return Values**

None

**Notes**

If an imported file type association is used to create a new configured file type association and the imported file type association is subsequently deleted, the configured file type association is not affected.

**Examples**

------------ EXAMPLE 1 ---------------

C:\PS> $dg = Get-BrokerDesktopGroup -Name "Sales VMs"
C:\PS> Remove-BrokerImportedFTA -DesktopGroupUids $dg.Uid
Deletes all imported file type associations belonging to the "Sales VMs" desktop group.
Remove-BrokerMachine

Oct 10, 2013
Removes one or more machines from its desktop group or catalog.

Syntax


Detailed Description

The Remove-BrokerMachine cmdlet removes one or more machines from their desktop group or catalog. There are three forms:

- Use the -InputObject parameter to remove a single machine instance or array of instances from their desktop group or catalog.
- Use the -MachineName parameter to remove the single named machine from its group or catalog.
- Use pipelining to pipe machine instances to the command.

To remove machines from their desktop group use the -DesktopGroup parameter; the specified group must be the one that contains the machines. If more than one machine is being removed from its group they must all be members of the same group.

If the -DesktopGroup parameter is not used then the machines are removed from their catalog. Removing a machine from its catalog deletes the record of the machine from the Citrix Broker Service.

Machines cannot be removed from their catalog while they are members of a desktop group.

Related topics

Add-BrokerMachine
Get-BrokerMachine

Parameters

-InputObject <Machine[]>
An array of machines to be removed from their desktop group or catalog.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>
-**MachineName**<String>

The name of the single machine to remove (must match the MachineName property of the machine).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**Force**<SwitchParameter>

Forces removal of machine from a desktop group even if it is still in use (that is, there are user sessions running on the machine). Forcing removal of a machine does not disconnect or logoff the user sessions.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**DesktopGroup**<DesktopGroup>

The desktop group from which the machines are to be removed, specified by name, UID, or instance.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
---|---
Accept Pipeline Input? | false

Input Type
Citrix.BrokerAdmin.SDK.Machine You can pipe in the machines to be removed.

Return Values
None

Examples

--------------- EXAMPLE 1 --------------------------
C:\PS> Remove-BrokerMachine -InputObject $machine -DesktopGroup $desktopGroup
C:\PS> Remove-BrokerMachine -InputObject $machine -DesktopGroup 2
C:\PS> Remove-BrokerMachine $machine -DesktopGroup "MyDesktopGroup"
These all remove a single machine from a desktop group, identifying the group by instance, UID, or name.

--------------- EXAMPLE 2 --------------------------
C:\PS> Remove-BrokerMachine -MachineName "DOMAIN\MyMachine" -DesktopGroup 2
C:\PS> Remove-BrokerMachine DOMAIN\MyMachine -DesktopGroup "MyDesktopGroup"
C:\PS> Remove-BrokerMachine DOMAIN\MyMachine -DesktopGroup $desktopGroup
These remove the machine called "DOMAIN\MyMachine" from its desktop group.

--------------- EXAMPLE 3 --------------------------
C:\PS> Remove-BrokerMachine -MachineName DOMAIN\MyMachine
C:\PS> Remove-BrokerMachine "DOMAIN\MyMachine"
C:\PS> Remove-BrokerMachine $machine
These all remove a machine from its catalog.

--------------- EXAMPLE 4 --------------------------
C:\PS> Get-BrokerMachine -Uid 3 | Remove-BrokerMachine -DesktopGroup $dg
C:\PS> Get-BrokerMachine -CatalogUid 4 | Remove-BrokerMachine
These find specific machines and remove them from their desktop group or catalog.
Remove-BrokerMachineCommand

Oct 10, 2013
Cancel a pending command queued for delivery to a desktop.

Syntax


Detailed Description

Sets the state of a pending command queued for delivery to a desktop to Canceled. The command is not removed from the system.

Related topics

Get-BrokerMachineCommand
New-BrokerMachineCommand

Parameters

- **InputObject** <MachineCommand[]>

Commands to cancel

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **LoggingId** <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress** <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
---|---
Accept Pipeline Input? | false

Input Type


Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------
Get-BrokerMachineCommand | Remove-BrokerMachineCommand
Cancel all pending commands.

-------------------------- EXAMPLE 2 --------------------------
Get-BrokerMachineCommand -Category "UPM" | Remove-BrokerMachineCommand
Cancel all pending commands that have the category "UPM".
Remove-BrokerMachineCommandMetadata

Oct 10, 2013
Deletes MachineCommand Metadata from the MachineCommand objects

Syntax

Remove-BrokerMachineCommandMetadata [-InputObject] <MachineCommand[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters>

Detailed Description

The Remove-BrokerMachineCommandMetadata cmdlet deletes Metadata from the MachineCommand objects.

Related topics

Parameters

-InputObject <MachineCommand[]>

Specifies the MachineCommand object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name <String>

Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
**-AdminAddress**: <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerMachineCommand You can pipe the MachineCommand to delete the metadata.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerMachineCommandMetadata -InputObject $obj-Uid -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for the MachineCommand whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerMachineCommandMetadata -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for all the MachineCommand in the site
Remove-BrokerMachineConfiguration

Oct 10, 2013
Deletes a machine configuration from the site or removes the association from a desktop group.

Syntax


Remove-BrokerMachineConfiguration [-Name] <String> [-DesktopGroup <DesktopGroup>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Remove-MachineConfiguration cmdlet deletes machine configurations from the site. A machine configuration cannot be removed while it is applied to a desktop group.

When a desktop group is provided, the Remove-MachineConfiguration cmdlet removes the association between machine configuration and the desktop group. In this case, the machine configuration is not removed from the site.

Related topics

New-BrokerMachineConfiguration
Get-BrokerMachineConfiguration
Set-BrokerMachineConfiguration
Rename-BrokerMachineConfiguration
Add-BrokerMachineConfiguration

Parameters

- **InputObject**<MachineConfiguration[]>
  Machine configuration to remove.

| Required? | true |
| Default Value | None |
| Accept Pipeline Input? | true (ByValue) |

- **Name**<String>
  Name of machine configuration for which the remove operation applies.

| Required? | true |
| Default Value | None |
Accept Pipeline Input? | true (ByPropertyName)

**-DesktopGroup**<DesktopGroup>

The desktop group from which this machine configuration is to be removed.

| Required? | false |
| Default Value | None |
| Accept Pipeline Input? | true (ByValue) |

**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**

Citrix.BrokerAdmin.SDK.MachineConfiguration Machine configuration to remove

**Return Values**

None

**Notes**

A machine configuration can only be removed if it is not currently applied to a desktop group.

**Examples**

-------------- EXAMPLE 1 --------------
Remove-BrokerMachineConfiguration -Name UPM\Finance
Removes the machine configuration named "UPM\Finance".

-------------------------- EXAMPLE 2 --------------------------

Remove-BrokerMachineConfiguration -Name Receiver\HumanResources -DesktopGroup SharedWorkers
Removes the association of the machine configuration named "Receiver\HumanResources" from the "SharedWorkers" desktop group.
Remove-BrokerMachineConfigurationMetadata

Oct 10, 2013
Deletes MachineConfiguration Metadata from the MachineConfiguration objects

Syntax

Remove-BrokerMachineConfigurationMetadata [-InputObject] <MachineConfiguration[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
The Remove-BrokerMachineConfigurationMetadata cmdlet deletes Metadata from the MachineConfiguration objects.

Related topics

Parameters

- **-InputObject** <MachineConfiguration[]>

Specifies the MachineConfiguration object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>

Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
-**AdminAddress**: `<String>`

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

`Citrix.BrokerAdmin.SDK.BrokerMachineConfiguration` You can pipe the `MachineConfiguration` to delete the metadata.

**Return Values**

None

**Examples**

------------- **EXAMPLE 1**  ---------------

C:\PS> Remove-BrokerMachineConfigurationMetadata -InputObject $obj-Uid -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for the `MachineConfiguration` whose instance is pointed by $obj-Uid

------------- **EXAMPLE 2**  ---------------

C:\PS> Remove-BrokerMachineConfigurationMetadata -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for all the `MachineConfiguration` in the site
Remove-BrokerMachineMetadata

Oct 10, 2013
Deletes Machine Metadata from the Machine objects

Syntax


Detailed Description

The Remove-BrokerMachineMetadata cmdlet deletes Metadata from the Machine objects.

Related topics

Parameters

- **-InputObject** <Machine[]>

Specifies the Machine object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>

Specifies the name of the Metadata to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**-AdminAddress <String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerMachine You can pipe the Machine to delete the metadata.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerMachineMetadata -InputObject $obj-Uid -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for the Machine whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerMachineMetadata -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for all the Machine in the site
Remove-BrokerPowerTimeScheme

Oct 10, 2013
Deletes an existing power time scheme.

Syntax

Remove-BrokerPowerTimeScheme [-InputObject] <PowerTimeScheme[]> [-LoggingId <Guid>][-AdminAddress <String>][<CommonParameters>]

Remove-BrokerPowerTimeScheme [-Name] <String> [-LoggingId <Guid>][-AdminAddress <String>][<CommonParameters>]

Detailed Description

The Remove-BrokerPowerTimeScheme cmdlet deletes a power time scheme from the system, and leaves the days that the time scheme used to cover for the associated desktop group as defaulting to all hours off-peak and all hours with pool size of -1.

Each power time scheme is associated with a particular desktop group, and covers one or more days of the week, defining which hours of those days are considered peak times and which are off-peak times. In addition, the time scheme defines a pool size value for each hour of the day for the days of the week covered by the time scheme. No one desktop group can be associated with two or more time schemes that cover the same day of the week.

For more information about the power policy mechanism and pool size management, see 'help about_Broker_PowerManagement'.

Related topics

Get-BrokerPowerTimeScheme
Set-BrokerPowerTimeScheme
New-BrokerPowerTimeScheme
Rename-BrokerPowerTimeScheme

Parameters

- **-InputObject**<PowerTimeScheme[]>
  The power time scheme to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name**<String>
  The name of the power time scheme to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-LoggingId**<Guid>
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
### Default Value

**Accept Pipeline Input?**

false

---

**-AdminAddress**

<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

**Required?**

false

**Default Value**

Localhost. Once a value is provided by any cmdlet, this value will become the default.

**Accept Pipeline Input?**

false

---

**Input Type**

Citrix.BrokerAdmin SDK.PowerTimeScheme The power time scheme to be deleted.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerPowerTimeScheme -Name 'Development Weekdays'

Deletes the power time scheme named 'Development Weekdays'.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerPowerTimeScheme -DesktopGroupUid (Get-BrokerDesktopGroup -name 'Finance desk1').Uid | Remove-BrokerPowerTimeScheme

Deletes all power time schemes for the desktop group named 'Finance desk1'.

---
Remove-BrokerPowerTimeSchemeMetadata

Oct 10, 2013
Deletes PowerTimeScheme Metadata from the PowerTimeScheme objects

Syntax

Remove-BrokerPowerTimeSchemeMetadata [-InputObject] <PowerTimeScheme[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Remove-BrokerPowerTimeSchemeMetadata cmdlet deletes Metadata from the PowerTimeScheme objects.

Related topics

Parameters

- **-InputObject** <PowerTimeScheme[]>

  Specifies the PowerTimeScheme object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>

  Specifies the name of the Metadata to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>
-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdmin.SDK.BrokerPowerTimeScheme You can pipe the PowerTimeScheme to delete the metadata.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerPowerTimeSchemeMetadata -InputObject $obj-Uid -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for the PowerTimeScheme whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerPowerTimeSchemeMetadata -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for all the PowerTimeScheme in the site
Remove-BrokerRebootCycleMetadata

Oct 10, 2013
Deletes RebootCycle Metadata from the RebootCycle objects

Syntax

Remove-BrokerRebootCycleMetadata [-InputObject] <RebootCycle[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Remove-BrokerRebootCycleMetadata cmdlet deletes Metadata from the RebootCycle objects.

Related topics

Parameters

- **-InputObject** <RebootCycle[]>

Specifies the RebootCycle object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>

Specifies the name of the Metadata to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerRebootCycle You can pipe the RebootCycle to delete the metadata.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerRebootCycleMetadata -InputObject $obj-Uid -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for the RebootCycle whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerRebootCycleMetadata -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for all the RebootCycle in the site
Remove-BrokerRebootSchedule

Oct 10, 2013
Removes the reboot schedule.

Syntax

Remove-BrokerRebootSchedule [-InputObject] <RebootSchedule[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-BrokerRebootSchedule [-DesktopGroupName] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Remove-BrokerRebootSchedule cmdlet is used to delete an existing reboot schedule.

Related topics

Get-BrokerRebootSchedule
Set-BrokerRebootSchedule
New-BrokerRebootSchedule
Stop-BrokerRebootCycle

Parameters

-InputObject <RebootSchedule[]>

The reboot schedule to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-DesktopGroupName <String>

The name of the desktop group whose reboot schedule is to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1265
Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.RebootSchedule Reboot schedules may be specified through pipeline input.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerRebootSchedule | Remove-BrokerRebootSchedule
Deletes every reboot schedule.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerRebootSchedule 12
Deletes the reboot schedule for the desktop group having Uid 12.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Remove-BrokerRebootSchedule -DesktopGroupName Accounting
Deletes the reboot schedule for the desktop group named Accounting.
Remove-BrokerRemotePCAccount

Oct 10, 2013
Delete RemotePCAccounts from the system.

Syntax


Detailed Description
Delete RemotePCAccounts from the site.

Related topics
Get-BrokerRemotePCAccount
New-BrokerRemotePCAccount
Set-BrokerRemotePCAccount

Parameters

-InputObject <RemotePCAccount[]>
Specifies the RemotePCAccounts to delete.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>
Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.
Required? | false  
---|---  
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.  
Accept Pipeline Input? | false  

Input Type

Citrix.BrokerAdmin.SDK.RemotePCAccount You can pipe the RemotePCAccounts to be deleted into this cmdlet.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Remove-BrokerRemotePCAccount 42
Delete RemotePCAccount 42.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-BrokerRemotePCAccount -OU 'any' | Remove-BrokerRemotePCAccount
Delete the 'any' OU RemotePCAccount.

-------------------------- EXAMPLE 3 --------------------------
C:\PS> Get-BrokerRemotePCAccount | Remove-BrokerRemotePCAccount
Delete all RemotePCAccounts.
Remove-BrokerScope

Oct 10, 2013
Remove the specified catalog/desktop group from the given scope(s).

Syntax

Remove-BrokerScope [-InputObject] <Scope[]> [-Catalog <Catalog>] [-DesktopGroup <DesktopGroup>] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters>

Detailed Description

The Remove-BrokerScope cmdlet is used to remove a catalog/desktop group object from the given scope(s).

To remove a catalog/desktop group from a scope you need permission to change the scopes of the catalog/desktop group.

If the catalog/desktop group is not in a specified scope, that scope will be silently ignored.

Related topics

Parameters

- **InputObject**: <Scope[]>

  Specifies the scopes to remove the object from.

  Required? | true
  --- | ---
  Default Value | null
  Accept Pipeline Input? | true (ByValue)

  **Catalog**: <Catalog>

  Specifies the catalog object to be removed.

  Required? | false
  --- | ---
  Default Value
  Accept Pipeline Input? | true (ByValue)

  **DesktopGroup**: <DesktopGroup>

  Specifies the desktop group object to be removed.

  Required? | false
  --- | ---
  Default Value

- `LoggingId`<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- `AdminAddress`<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdminSDK.Scope You can pipe scopes to Remove-BrokerScope.

Examples

-------------------- EXAMPLE 1 -------------------------

C:\PS> Remove-BrokerScope -Scope Chalfont,Cambourne -DesktopGroup "Win7 Desktops"
Removes the "Win7 Desktops" desktop group from both the Chalfont and Cambourne scopes.

-------------------- EXAMPLE 2 -------------------------

C:\PS> 'Chalfont','Cambourne' | Remove-BrokerScope -DesktopGroup 'Win7 Desktops'
Removes the "Win7 Desktops" desktop group from both the Chalfont and Cambourne scopes.
Remove-BrokerSessionMetadata

Oct 10, 2013
Deletes Session Metadata from the Session objects

Syntax

Remove-BrokerSessionMetadata [-InputObject] <Session[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

The Remove-BrokerSessionMetadata cmdlet deletes Metadata from the Session objects.

Related topics

Parameters

- **-InputObject** <Session[]>
  Specifies the Session object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>
  Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
**-AdminAddress**: String

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerSession You can pipe the Session to delete the metadata.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerSessionMetadata -InputObject $obj-Uid -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for the Session whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerSessionMetadata -Name "MyMetadataName"

This command deletes the Metadata "MyMetadataName" key-value pair for all the Session in the site
Remove-BrokerSiteMetadata

Oct 10, 2013
Deletes Site Metadata from the Site objects

Syntax

Remove-BrokerSiteMetadata -Name <String> [-InputObject <Site[]>] [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

The Remove-BrokerSiteMetadata cmdlet deletes Metadata from the Site objects.

Related topics

Parameters

- **-Name**<String>

  Specifies the name of the Metadata to be deleted

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-InputObject**<Site[]>

  Specifies the Site object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-LoggingId**<Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerSite You can pipe the Site to delete the metadata.

**Return Values**

None

**Examples**

```
-------------------------- EXAMPLE 1 --------------------------
C:\PS> Remove-BrokerSiteMetadata -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for the Site
```
Remove-BrokerTag

Oct 10, 2013
Removes a tag from the system or clears a tag to object association.

Syntax


Remove-BrokerTag [-Name] <String> [-Desktop <Desktop>] [-DesktopGroup <DesktopGroup>] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Clears tag to object associations or delete tags from the system altogether.

To clear an association, supply one of the Application, DesktopGroup or PrivateDesktop parameters.

To delete the tag, and any associations between the tag and other objects in the database, specify the tag without any associated object parameters.

Related topics

Add-BrokerTag
Get-BrokerTag
New-BrokerTag
Rename-BrokerTag

Parameters

-InputObject <Tag[]>

Specifies one or more tag objects.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name <String>

Specifies a tag by name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
- **Desktop**<Desktop>

Cleans the association between the given tag and Desktop.

| Required? | false |
| Default Value |

- **DesktopGroup**<DesktopGroup>

Cleans the association between the given tag and desktop group.

| Required? | false |
| Default Value |

- **LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value |

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value |

Input Type

Citrix.BrokerAdmin.SDK.Tag Tags may be specified through pipeline input.

Return Values
None

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerTag $tag -Desktop $desktop
Clears the association between a tag and a desktop. Note that the tag itself continues to exist as an object in the database.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerTag $tag
Deletes the tag object from the database and clears any associations that may exist between that tag and other objects.
Remove-BrokerTagMetadata

Oct 10, 2013
Deletes Tag Metadata from the Tag objects

Syntax

Remove-BrokerTagMetadata [-InputObject] <Tag[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Remove-BrokerTagMetadata cmdlet deletes Metadata from the Tag objects.

Related topics

Parameters

- **-InputObject**<Tag[]>

Specifies the Tag object's instance whose Metadata is to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

- **-Name**<String>

Specifies the name of the Metadata to be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

- **-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerTag You can pipe the Tag to delete the metadata.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-BrokerTagMetadata -InputObject $obj-Uid -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for the Tag whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-BrokerTagMetadata -Name "MyMetadataName"
This command deletes the Metadata "MyMetadataName" key-value pair for all the Tag in the site
Remove-BrokerUser

Oct 10, 2013
Remove broker user objects from another broker object

Syntax


Detailed Description

The Remove-BrokerUser cmdlet removes broker user objects from another specified object, such as a broker private desktop, to which the user had previously been added.

Related topics

Add-BrokerUser
Get-BrokerUser

Parameters

-**InputObject** <User[]>

Specifies the user objects to remove.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-**Name** <String>

Specifies the user objects to remove, based on their Name property.

| Required? | true |
| Default Value | null |
| Accept Pipeline Input? | true (ByPropertyName) |

-**Application** <Application>

The application from which to remove the user

| Required? | false |
- **Machine**<br>The machine from which to remove the user

| Required? | false |
| Default Value | null |
| Accept Pipeline Input? | true (ByValue) |

- **PrivateDesktop**<br>The desktop from which to remove the user

| Required? | false |
| Default Value | null |
| Accept Pipeline Input? | true (ByValue) |

- **LoggingId**<br>Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **AdminAddress**<br>Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |
Citrix.BrokerAdmin.SDK.Usuario You can pipe the users to be removed to Remove-BrokerUser.

Return Values
None

Notes
Specify one of the -Machine or -PrivateDesktop parameters only.

Examples
-------------------------- EXAMPLE 1 --------------------------
Remove-BrokerUser "DOMAIN\UserName" -PrivateDesktop "DOMAIN\MachineName"
Remove the assignment of the specified private desktop to the specified user.
Rename-BrokerAccessPolicyRule

Oct 10, 2013
Renames a rule in the site's access policy.

Syntax


Detailed Description

The Rename-BrokerAccessPolicyRule cmdlet renames a rule in the site's access policy. The Name property of the rule is changed.

An access policy rule defines a set of connection filters and access control rights relating to a desktop group. These allow fine-grained control of what access is granted to a desktop group based on details of, for example, a user's endpoint device, its address, and the user's identity.

Related topics

New-BrokerAccessPolicyRule
Get-BrokerAccessPolicyRule
Set-BrokerAccessPolicyRule
Remove-BrokerAccessPolicyRule

Parameters

- **-InputObject** <AccessPolicyRule[]>

  The access policy rule to be renamed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

- **-Name** <String>

  The existing name of the access policy rule to be renamed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
- **NewName**<String>

  The new name for the access policy rule being renamed. The new name must not match that of any other existing rules in the policy.
  
  | Required? | true |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **PassThru**<SwitchParameter>

  This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.
  
  | Required? | false |
  | Default Value | False |
  | Accept Pipeline Input? | false |

- **LoggingId**<Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.
  
  | Required? | false |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.
  
  | Required? | false |
  | Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
  | Accept Pipeline Input? | false |

---

Input Type

Return Values

None or Citrix.BrokerAdmin.SDK.AccessPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.AccessPolicyRule object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-BrokerAccessPolicyRule 'Sales' -NewName 'TeleSales'
Renames the access policy rule called Sales to TeleSales. The new name of the rule must be unique in the access policy.
Rename-BrokerAppAssignmentPolicyRule

Oct 10, 2013
Renames an application rule in the site's assignment policy.

Syntax


Detailed Description

The Rename-BrokerAppAssignmentPolicyRule cmdlet renames an application rule in the site's assignment policy. The Name property of the rule is changed.

An application rule in the assignment policy defines the users who are entitled to a self-service persistent machine assignment from the rule's desktop group; once assigned the machine can run one or more applications published from the group.

Related topics

New-BrokerAppAssignmentPolicyRule
Get-BrokerAppAssignmentPolicyRule
Set-BrokerAppAssignmentPolicyRule
Remove-BrokerAppAssignmentPolicyRule

Parameters

-InputObject <AppAssignmentPolicyRule[]>

The application rule in the assignment policy to be renamed.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-Name <String>

The existing name of the application rule in the assignment policy to be renamed.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | true (ByPropertyName)

-NewName<String>
The new name of the application rule in the assignment policy being renamed. The new name must not match that of any other existing rule in the policy (irrespective of whether it is a desktop or application rule).

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-PassThru<SwitchParameter>
This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

-LoggingId<Guid>
Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AdminAddress<String>
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

Input Type
Return Values

None or Citrix.BrokerAdmin.SDK.AppAssignmentPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.AppAssignmentPolicyRule object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-BrokerAppAssignmentPolicyRule 'Offshore' -NewName 'Remote Workers'
Renames the application rule in the assignment policy called Offshore to Remote Workers. The new name of the rule must be unique in the assignment policy.
Rename-BrokerAppEntitlementPolicyRule

Oct 10, 2013
Renames an application rule in the site’s entitlement policy.

Syntax


Detailed Description

The Rename-BrokerAppEntitlementPolicyRule cmdlet renames an application rule in the site’s entitlement policy. The Name property of the rule is changed.

An application rule in the entitlement policy defines the users who are allowed per-session access to a machine to run one or more applications published from the rule’s desktop group.

Related topics

New-BrokerAppEntitlementPolicyRule
Get-BrokerAppEntitlementPolicyRule
Set-BrokerAppEntitlementPolicyRule
Remove-BrokerAppEntitlementPolicyRule

Parameters

- **-InputObject** <AppEntitlementPolicyRule[]>

The application rule in the entitlement policy to be renamed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>

The existing name of the application rule in the entitlement policy to be renamed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
- **NewName**<String>

  The new name of the application rule in the entitlement policy being renamed. The new name must not match that of any other existing rule in the policy (irrespective of whether it is a desktop or application rule).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **PassThru**<SwitchParameter>

  This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values

None or Citrix.BrokerAdmin.SDK.AppEntitlementPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.AppEntitlementPolicyRule object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-BrokerAppEntitlementPolicyRule 'Prod Dev' -NewName 'Product Development'

Renames the application rule in the entitlement policy called Prod Dev to Product Development. The new name of the rule must be unique in the entitlement policy.
Rename-BrokerApplication

Oct 10, 2013
Renames an application.

Syntax


Rename-BrokerApplication [-Name] <String> [-NewName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Rename-BrokerApplication cmdlet changes the administrative name of an application. An application cannot have the same name as another application.

Renaming an application does not alter its published name. To change the name with which this application appears to end-users, set a new value for the PublishedName property using the Set-BrokerApplication cmdlet.

Renaming an application does not alter its BrowserName. If the BrowserName property also needs to be changed, use the Set-BrokerApplication cmdlet to modify it.

Related topics

New-BrokerApplication
Set-BrokerApplication
Get-BrokerApplication
Remove-BrokerApplication

Parameters

-InputObject <Application[]>

Specifies the application to rename.

| Required? | true |
| Default Value | null |
| Accept Pipeline Input? | true (ByValue) |

-Name <String>

Specifies the name of the application to rename.

<p>| Required? | true |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>null</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
<tr>
<td><strong>NewName</strong>&lt;String&gt;</td>
<td>Specifies the new name for the application.</td>
</tr>
<tr>
<td>Required?</td>
<td>true</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>PassThru</strong>&lt;SwitchParameter&gt;</td>
<td>This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>LoggingId</strong>&lt;Guid&gt;</td>
<td>Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>AdminAddress</strong>&lt;String&gt;</td>
<td>Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Citrix.BrokerAdmin.SDK.Application you can pipe applications to Rename-BrokerApplication.

Return Values

None or Citrix.BrokerAdmin.SDK.Application

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.Application object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-BrokerApplication -Name "Old Name" -NewName "New Name"
Renames the application with name "Old Name" to "New Name".

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerApplication -Uid 1 | Rename-BrokerApplication -NewName "New Name" -PassThru
Renames application with the Uid 1 to "New Name", showing the result.
Rename-BrokerAssignmentPolicyRule

Oct 10, 2013
Renames a desktop rule in the site's assignment policy.

Syntax


Detailed Description

The Rename-BrokerAssignmentPolicyRule cmdlet renames a desktop rule in the site's assignment policy. The Name property of the rule is changed.

A desktop rule in the assignment policy defines the users who are entitled to self-service persistent machine assignments from the rule's desktop group. A rule defines how many machines a user is allowed from the group for delivery of full desktop sessions.

Related topics

New-BrokerAssignmentPolicyRule
Get-BrokerAssignmentPolicyRule
Set-BrokerAssignmentPolicyRule
Remove-BrokerAssignmentPolicyRule

Parameters

-**InputObject** <AssignmentPolicyRule[]>

The desktop rule in the assignment policy to be renamed.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-**Name** <String>

The existing name of the desktop rule in the assignment policy to be renamed.

| Required? | true |
| Default Value | |
Accept Pipeline Input? | true (ByPropertyName)
---|---

**-NewName<String>**

The new name of the desktop rule in the assignment policy being renamed. The new name must not match that of any other existing rule in the policy (irrespective of whether it is a desktop or application rule).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-PassThru<SwitchParameter>**

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId<Guid>**

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Return Values

None, or Citrix.BrokerAdmin.SDK.AssignmentPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.AssignmentPolicyRule object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-BrokerAssignmentPolicyRule 'Offshore' -NewName 'Remote Workers'

Renames the desktop rule in the assignment policy called Offshore to Remote Workers. The new name of the rule must be unique in the assignment policy.
Rename-BrokerCatalog

Oct 10, 2013
Renames a catalog.

Syntax

Rename-BrokerCatalog [-InputObject] <Catalog[]> [-NewName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Rename-BrokerCatalog [-Name] <String> [-NewName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Rename-BrokerCatalog cmdlet changes the name of a catalog. A catalog cannot have the same name as another catalog.

The following special characters are not allowed in a catalog name: \\ / : # * ? = < > | ()"'

Related topics

Get-BrokerCatalog
New-BrokerCatalog
Remove-BrokerCatalog
Set-BrokerCatalog

Parameters

-InputObject <Catalog[]>
Specifies the catalog to rename.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name <String>
Specifies the name of the catalog to rename.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByPropertyName)</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
- **NewName**<String>

Specifies the new name of the catalog.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdmin.SDK.Catalog You can pipe catalogs to Rename-BrokerCatalog.

Return Values
None or Citrix.BrokerAdmin.SDK.Catalog

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.Catalog object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-BrokerCatalog -Name "Old Name" -NewName "New Name"
Renames the catalog with the name "Old Name" to "New Name".

-------------------------- EXAMPLE 2 --------------------------

C:\PS> c:\$catalog = Get-BrokerCatalog -Name "Old Name"
C:\PS> Rename-BrokerCatalog -InputObject $catalog -NewName "New Name"
Renames the catalog with the name "Old Name" to "New Name".
Rename-BrokerDesktopGroup

Renames a desktop group.

Syntax


Rename-BrokerDesktopGroup [-Name] <String> [-NewName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Rename-BrokerDesktopGroup cmdlet changes the name of a desktop group. A desktop group cannot have the same name as another desktop group.

Related topics

Get-BrokerDesktopGroup

New-BrokerDesktopGroup

Set-BrokerDesktopGroup

Remove-BrokerDesktopGroup

Parameters

-InputObject <DesktopGroup[]>

Specifies the desktop group to rename.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>null</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name <String>

Specifies the name of the desktop group to rename.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>null</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-NewName <String>
Specifies the new name that the desktop group will have.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.DesktopGroup You can pipe desktop groups to Rename-BrokerDesktopGroup.

**Return Values**

None or Citrix.BrokerAdmin.SDK.DesktopGroup
This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a `Citrix.BrokerAdmin.SDK.DesktopGroup` object.

Notes

Renaming a desktop group does not alter its published name. If you need to change the name with which this desktop group appears to end-users, set a new value for the PublishedName property using the Set-BrokerDesktopGroup cmdlet.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-BrokerDesktopGroup -Name "Old Name" -NewName "New Name"
Renames desktop group with the name "Old Name" to "New Name".

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerDesktopGroup -Uid 1 | Rename-BrokerDesktopGroup -NewName "New Name" -PassThru
Renames desktop group with the Uid 1 to "New Name", showing the result.
Rename-BrokerEntitlementPolicyRule

Oct 10, 2013
Renames a desktop rule in the site's entitlement policy.

Syntax

```

```

Detailed Description

The Rename-BrokerEntitlementPolicyRule cmdlet renames a desktop rule in the site's entitlement policy. The Name property of the rule is changed.

A desktop rule in the entitlement policy defines the users who are allowed per-session access to a machine from the rule's associated desktop group to run a full desktop session.

Related topics

- New-BrokerEntitlementPolicyRule
- Get-BrokerEntitlementPolicyRule
- Set-BrokerEntitlementPolicyRule
- Remove-BrokerEntitlementPolicyRule

Parameters

- **-InputObject** `<EntitlementPolicyRule[]>`

  The desktop rule in the entitlement policy to be renamed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** `<String>`

  The existing name of the desktop rule in the entitlement policy to be renamed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
### -NewName

The new name of the desktop rule in the entitlement policy being renamed. The new name must not match that of any other existing rule in the policy (irrespective of whether it is a desktop or application rule).

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

### -PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

### -LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -AdminAddress

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |
Return Values

None or Citrix.BrokerAdmin.SDK.EntitlementPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.EntitlementPolicyRule object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-BrokerEntitlementPolicyRule 'Prod Dev' -NewName 'Product Development'
Renames the desktop rule in the entitlement policy called Prod Dev to Product Development. The new name of the rule must be unique in the entitlement policy.
**Rename-BrokerMachineConfiguration**

Oct 10, 2013
Renames a machine configuration.

**Syntax**

```powershell

```

**Detailed Description**

The `Rename-MachineConfiguration` cmdlet changes the name of a machine configuration. A machine configuration cannot have the same name as another machine configuration associated with the same slot.

**Related topics**

- `New-BrokerMachineConfiguration`
- `Get-BrokerMachineConfiguration`
- `Set-BrokerMachineConfiguration`
- `Remove-BrokerMachineConfiguration`
- `Add-BrokerMachineConfiguration`

**Parameters**

**-InputObject** `<MachineConfiguration[]>`

Machine configuration to rename.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>None</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-Name** `<String>`

Current name of machine configuration.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>None</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
- **NewName**:<String>

New name for machine configuration. This may have the form "ConfigurationSlotName\MachineConfigurationName" or "MachineConfigurationName". If the "ConfigurationSlotName" is provided it must match the name of the configuration slot that the machine configuration is associated with.

| Required? | true |
| Default Value | None |
| Accept Pipeline Input? | false |

- **PassThru**:<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

- **LoggingId**:<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Accept Pipeline Input? | false |

- **AdminAddress**:<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**

Return Values
None or Citrix.BrokerAdmin.SDK.MachineConfiguration

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.MachineConfiguration object.

Notes
The configuration slot can not be changed. Thus the left term of the Name and NewName must match.

Examples

-------------------------- EXAMPLE 1 --------------------------

Rename-BrokerMachineConfiguration -Name "UPM\All Departments" -NewName "UPM\Finance Department"
Renames the machine configuration named "UPM\All Departments" to "UPM\Finance Department".

-------------------------- EXAMPLE 2 --------------------------

Rename-BrokerMachineConfiguration -Name "UPM\All Departments" -NewName "Finance Department"
Renames the machine configuration named "UPM\All Departments" to "UPM\Finance Department".
Rename-BrokerPowerTimeScheme

Oct 10, 2013
Changes the name of an existing power time scheme.

Syntax


Detailed Description

The Rename-BrokerPowerTimeScheme cmdlet renames a particular power time scheme.

Each power time scheme is associated with a particular desktop group, and covers one or more days of the week, defining which hours of those days are considered peak times and which are off-peak times. In addition, the time scheme defines a pool size value for each hour of the day for the days of the week covered by the time scheme. No one desktop group can be associated with two or more time schemes that cover the same day of the week.

For more information about the power policy mechanism and pool size management, see ‘help about_Broker_PowerManagement’.

Related topics

Get-BrokerPowerTimeScheme
Set-BrokerPowerTimeScheme
New-BrokerPowerTimeScheme
Remove-BrokerPowerTimeScheme

Parameters

- **InputObject** <PowerTimeScheme[]>

  The power time scheme to be renamed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name** <String>

  The current name of the power time scheme to be renamed.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
- **NewName**: <String>

  The new name to be applied to the power time scheme.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **PassThru**: <SwitchParameter>

  This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**: <Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**: <String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Citrix.BrokerAdmin.SDK.PowerTimeScheme The power time scheme to be renamed.

Return Values
None or Citrix.BrokerAdmin.SDK.PowerTimeScheme

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.PowerTimeScheme object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-BrokerPowerTimeScheme -Name 'Development Weekdays' -NewName 'Dev Week'
Renames the power time scheme named 'Development Weekdays' to 'Dev Week'.
Rename-BrokerTag

Oct 10, 2013
Rename one or more tags.

Syntax

Rename-BrokerTag [-InputObject] <Tag[]> [-NewName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Rename-BrokerTag [-Name] <String> [-NewName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Renames one or more tags with the supplied name.

Related topics

Add-BrokerTag

Get-BrokerTag

New-BrokerTag

Remove-BrokerTag

Parameters

-InputObject <Tag[]>

Specifies one or more tag objects to rename.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name <String>

Identifies tags to be renamed by name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-NewName <String>

Specifies new name for the tags.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
### Required?
<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
</table>

### Default Value

### Accept Pipeline Input?
| Accept Pipeline Input? | false |

### -PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>

### -LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>

### -AdminAddress

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |

### Input Type

Citrix.BrokerAdmin.SDK.Tag The tag to rename can be piped into this cmdlet.

### Return Values

None or Citrix.BrokerAdmin.SDK.Tag

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.Tag object.
Notes

Note that when renaming a tag, its UUID remains the same and any associations are maintained.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-BrokerTag -Name 'OldName' -NewName 'ReplacementName'
Renames tags with the name 'OldName' to 'ReplacementName'.
Reset-BrokerLicensingConnection

Oct 10, 2013
Resets the broker's license server connection.

Syntax

```
Reset-BrokerLicensingConnection [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]
```

Detailed Description

The Reset-BrokerLicensingConnection cmdlet resets the broker's connection to the license server.

Licensing changes resulting from new license files or alterations to the site-level licensing properties don't become effective immediately. There will typically be a delay as the changes are propagated across the site based on the scheduling of refresh logic built into the controllers and the license server.

Resetting the connection causes the list of available licenses for the connection to be updated. After adding licenses or changing the site-level licensing properties you can run Reset-BrokerLicensingConnection to ensure that the broker can access the new licenses immediately.

Each broker service instance holds its own connection to the license server. In order for the licensing changes to be applied immediately throughout the XenDesktop site this command needs to be run on every controller in the site.

Related topics

Get-BrokerSite
Set-BrokerSite

Parameters

- **-LoggingId**<Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress**<String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

None

Examples

------------------------------- EXAMPLE 1 -----------------------------

C:\PS> Reset-BrokerLicensingConnection
Reset the broker's license server connection.
Reset-BrokerServiceGroupMembership

Oct 10, 2013
Makes the broker load, from the specified Configuration Service instance, the addresses that the Service can be reached on.

Syntax

Reset-BrokerServiceGroupMembership -ConfigServiceInstance <PSObject[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

This cmdlet reloads the access permissions and Configuration Service locations for the Broker Service. It must be run on at least one instance of the Broker Service after installation and registration with the Configuration Service. Without this operation, the Broker Services cannot communicate with the other services that form part of the Citrix XenDesktop deployment. Once the cmdlet is run, the services keep up to date when additional services are added to the deployment provided that the Configuration Service is not stopped. Run this cmdlet to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one Configuration Service instance is passed to the cmdlet, only the first is used that meets the expected service type requirements.

Related topics

Get-BrokerServiceInstance
Get-BrokerServiceStatus
Get-ConfigRegisteredServiceInstance

Parameters

-**ConfigServiceInstance**<PSObject[]>

The Configuration Service Instance object that represents a service instance of the type InterService that in turn references a Configuration Service for the deployment.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>LocalHost</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.
Required?  |  false
--- | ---
Default Value  |  Localhost. Once a value is provided by any cmdlet, this value will become the default.
Accept Pipeline Input?  |  false

### Input Type

Citrix.XDPowerShell.ServiceInstance You can pipe a ServiceInstance that contains a ServiceInstance object that in turn refers to the Configuration Service InterService interface.

### Return Values

Citrix.XDPowerShell.ServiceInstance

Reset-BrokerServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the Broker Service instance.

### Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS$gt;Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-BrokerServiceGroupMembership
Resets the Broker Service's group membership. This assumes that the Configuration Service is running and configured on the same machine as the Broker service.

-------------------------- EXAMPLE 2 --------------------------

c:\PS$gt;Get-ConfigRegisteredServiceInstance -ServiceType Config -AdminAddress OtherServer.com | Reset-BrokerServiceGroupMembership
Resets the Broker Service's group membership. This uses a Configuration Service that is running and configured on a server called OtherServer.com.
Send-BrokerSessionMessage

Oct 10, 2013
Sends a message to a session.

Syntax

```
Send-BrokerSessionMessage [-InputObject] <Session[]> [-MessageStyle] <SendMessageStyle> [-Title] <String> [-Text] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]
```

Detailed Description

Generates a message box in the target session(s).

Related topics

Get-BrokerSession

Parameters

- **-InputObject** `<Session[]>`

The target session(s) to send the message to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-MessageStyle** `<SendMessageStyle>`

The style of message box to use (valid values are Critical, Question, Exclamation, or Information).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Title** `<String>`

Text to display in the messagebox title bar.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Text** `<String>`
The message to display.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-LoggingId<Guid>**

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.Session The session to which to send the message can be piped in.

**Return Values**

None

**Notes**

Sessions can be passed as the InputObject parameter as either session objects or their numeric Uids.

This operation is non-blocking and returns before it completes. The operation, however, is unlikely to fail unless there are communication problems between controller and machine, if bad arguments are passed to the cmdlet itself or if the machine cannot successfully execute the operation.

The transient nature of sessions means that the list of session objects or UIDs supplied to Send-BrokerSessionMessage could consist of valid and invalid sessions. Invalid sessions are detected and disregarded and the send message operation is invoked on the machines running valid sessions.

The system can fail to invoke the operation if the machine is not in an appropriate state or if there are problems in communicating with the machine. When an operation is invoked the system detects if the operation was initiated successfully...
or not by the session. As this operation is non-blocking the system doesn't detect or report whether the operation ultimately succeeded or failed after its successful initialization in the session.

Operation failures are reported through the broker SDK error handling mechanism (see about_Broker_ErrorHandling). In the event of errors the SdkErrorRecord error status code is set to SessionOperationFailed and its error data dictionary is populated with the following entries:

- OperationsAttemptedCount: The number of operations attempted.
- OperationsFailedCount - The number of failed operations.
- OperationsSucceededCount - The number of successfully executed operations.
- UnresolvedSessionFailuresCount - The number of operations that failed due to invalid sessions being supplied.
- OperationInvocationFailuresCount - The number of operations that failed because they could not be invoked in the session.
- DesktopExecutionFailuresCount - The number of operations that failed because they could not be successfully executed in the session.

The SdkErrorRecord message will also display the number of attempted, failed and successful operations in the following format:


Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> $sessions = Get-BrokerSession -UserName MYDOMAIN\*
C:\PS> Send-BrokerSessionMessage $sessions -MessageStyle Information -Title TestTitle -Text TestMessage
Sends a message to all the sessions for any user in MYDOMAIN.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> $desktop = Get-BrokerDesktop -Uid 1
C:\PS> Send-BrokerSessionMessage $desktop.SessionUid -MessageStyle Information -Title TestTitle -Text TestMessage
Sends a message to the session on the desktop with Uid 1.

-------------------------- EXAMPLE 3 --------------------------
C:\PS> trap [Citrix.Broker.Admin.SDK.SdkOperationException]
C:\PS> {
C:\PS>   write ($"Exception name = " + $_.Exception.GetType().FullName)
C:\PS>   write ($"SdkOperationException.Status = " + $_.Exception.Status)
C:\PS>   write ($"SdkOperationException.ErrorData=")
C:\PS>   $_.Exception.ErrorData
C:\PS>   write ($"SdkOperationException.InnerException = " + $_.Exception.InnerException)
C:\PS>   $_.Exception.InnerException
C:\PS>   continue
C:\PS> }
C:\PS>
C:\PS> Send-BrokerSessionMessage -InputObject 10,11,12 -MessageStyle Information -Title "message title" -Text "message text"
Trap and display error information.
Set-BrokerAccessPolicyRule

Oct 10, 2013

Modifies an existing rule in the site's access policy.

Syntax


Detailed Description

The Set-BrokerAccessPolicyRule cmdlet modifies an existing rule in the site's access policy.

An access policy rule defines a set of connection filters and access control rights relating to a desktop group. These allow fine-grained control of what access is granted to a desktop group based on details of, for example, a user's endpoint device, its address, and the user's identity.

Changing a rule does not affect existing user sessions, but it may result in users being unable to launch new sessions, or reconnect to disconnected sessions if the change removes access to the desktop group delivering those sessions.

Related topics

New-BrokerAccessPolicyRule
Get-BrokerAccessPolicyRule
Rename-BrokerAccessPolicyRule
Remove-BrokerAccessPolicyRule

Parameters

-InputObject <AccessPolicyRule[]>

The access policy rule to be modified.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | true (ByValue)
---|---
-Name | <String>
The name of the access policy rule to be modified.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-PassThru | <SwitchParameter>
This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

-AddExcludedClientIPs | <IPAddressRange[]>
Adds the specified user device IP addresses to the excluded client IP address filter of the rule.
See the ExcludedClientIPs parameter for more information.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AddExcludedClientNames | <String[]>
Adds the specified user device names to the excluded client names filter of the rule.
See the ExcludedClientNames parameter for more information.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AddExcludedSmartAccessTags | <String[]>
Adds the specified SmartAccess tags to the excluded SmartAccess tags filter of the rule.
See the ExcludedSmartAccessTags parameter for more information.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AddExcludedUsers | <User[]>
Adds the specified users and groups to the excluded users filter of the rule.
See the ExcludedUsers parameter for more information.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-AddIncludedClientIPs</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-AddIncludedClientNames</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-AddIncludedSmartAccessTags</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-AddIncludedUsers</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-AllowedConnections</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-AddIncludedClientIPs

Adds the specified user device IP addresses to the included client IP address filter of the rule.

See the IncludedClientIPs parameter for more information.

-AddIncludedClientNames

Adds the specified user device names to the included client names filter of the rule.

See the IncludedClientNames parameter for more information.

-AddIncludedSmartAccessTags

Adds the specified SmartAccess tags to the included SmartAccess tags filter of the rule.

See the IncludedSmartAccessTags parameter for more information.

-AddIncludedUsers

Adds the specified users and groups to the included users filter of the rule.

See the IncludedUsers parameter for more information.

-AllowedConnections

Changes whether connections must be local or via Access Gateway, and if so whether specified SmartAccess tags must be provided by Access Gateway with the connection. This property forms part of the included SmartAccess tags filter.

Valid values are Filtered, NotViaAG, and ViaAG.
For a detailed description of this property see "help about_Broker_AccessPolicy".

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Required?</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-**AllowedProtocols**<String[]>

Changes the protocols (for example HDX, RDP) available to the user for sessions delivered from the rule's desktop group. If the user gains access to a desktop group by multiple rules, the allowed protocol list is the combination of the protocol lists from all those rules.

If the protocol list is empty, access to the desktop group is implicitly denied.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Required?</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-**AllowedUsers**<AllowedUser>

Changes the behavior of the included users filter of the rule. This can restrict access to a list of named users or groups, or allow access to any authenticated user. For a detailed description of this property see "help about_Broker_AccessPolicy".

Valid values are Filtered, AnyAuthenticated, and Any.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Required?</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-**AllowRestart**<Boolean>

Changes whether the user can restart sessions delivered from the rule's desktop group. Session restart is handled as follows: For sessions on single-session power-managed machines, the machine is powered off, and a new session launch request made; for sessions on multi-session machines, a logoff request is issued to the session, and a new session launch request made; otherwise the property is ignored.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Required?</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-**Description**<String>

Changes the description of the rule. The text is purely informational for the administrator, it is never visible to the end user.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Required?</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-**Enabled**<Boolean>

Changes whether the rule is enabled or disabled. A disabled rule is ignored when evaluating the site's access policy.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Required?</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>-ExcludedClientIPFilterEnabled</strong></td>
<td>Boolean</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>-ExcludedClientIPs</strong></td>
<td>IPAddressRange[]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>-ExcludedClientNameFilterEnabled</strong></td>
<td>Boolean</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>-ExcludedClientNames</strong></td>
<td>String[]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>-ExcludedSmartAccessFilterEnabled</strong></td>
<td>Boolean</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>-ExcludedSmartAccessTags</strong></td>
<td>String[]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
-**ExcludedUserFilterEnabled**<Boolean>

Changes whether the excluded users filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**ExcludedUsers**<User[]>

Changes which users and groups are explicitly denied access to the rule's desktop group. This property forms part of the excluded users filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**HdxSslEnabled**<Boolean>

Indicates whether SSL encryption is enabled for sessions delivered from the rule's desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**IncludedClientIPFilterEnabled**<Boolean>

Changes whether the included client IP address filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**IncludedClientIPs**<IPAddressRange[]>

Changes which IP addresses of user devices allowed access to the rule's desktop group. Addresses can be specified as simple numeric addresses or as subnet masks (for example, 10.40.37.5 or 10.40.0.0/16). This property forms part of the included client IP address filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**IncludedClientNameFilterEnabled**<Boolean>

Changes whether the included client name filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

-**IncludedClientNames**<String[]>

Changes which client names of user devices allowed access to the rule's desktop group. Client names can be specified as full names (for example, 'Citrix Server 1', 'John Doe'). This property forms part of the included client name filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**-IncludedClientNames<String[]>**

Changes which names of user devices are allowed access to the rule's desktop group. This property forms part of the included client names filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-IncludedSmartAccessFilterEnabled<Boolean>**

Changes whether the included SmartAccess tags filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-IncludedSmartAccessTags<String[]>**

Changes which SmartAccess tags grant access to the rule's desktop group if any occur in those provided by Access Gateway with the user's connection. This property forms part of the excluded SmartAccess tags filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-IncludedUserFilterEnabled<Boolean>**

Changes whether the included users filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-IncludedUsers<User[]>**

Changes which users and groups are granted access to the rule's desktop group. This property forms part of the included users filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-RemoveExcludedClientIPs<IPAddressRange[]>**

Removes the specified user device IP addresses from the excluded client IP address filter of the rule.

See the ExcludedClientIPs parameter for more information.
### -RemoveExcludedClientNames <String[]>

Removes the specified user device names from the excluded client names filter of the rule.

See the ExcludeClientNames parameter for more information.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -RemoveExcludedSmartAccessTags <String[]>

Removes the specified SmartAccess tags from the excluded SmartAccess tags filter of the rule.

See the ExcludeSmartAccessTags parameter for more information.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -RemoveExcludedUsers <User[]>

Removes the specified users and groups from the excluded users filter of the rule.

See the ExcludeUsers parameter for more information.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -RemoveIncludedClientIPs <IPAddressRange[]>

Removes the specified user device IP addresses from the included client IP address filter of the rule.

See the IncludeClientIPs parameter for more information.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -RemoveIncludedClientNames <String[]>

Removes the specified client names from the included client names filter of the rule.

See the IncludeClientNames parameter for more information.

| Required? | false |
- **RemoveIncludedSmartAccessTags**<String[]>

Removes the specified SmartAccess tags from the included SmartAccess tags filter of the rule.

See the IncludedSmartAccessTags parameter for more information.

- **RemoveIncludedUsers**<User[]>

Removes the specified users and groups from the included users filter of the rule.

See the IncludedUsers parameter for more information.

- **LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

---

**Input Type**

Citrix.BrokerAdmin.SDK.AccessPolicyRule The access policy rule to be modified.

**Return Values**

None, or Citrix.BrokerAdmin.SDK.AccessPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.AccessPolicyRule object.

**Examples**
-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerAccessPolicyRule 'Temp Staff' -AddIncludedUsers office\contractors

Adds user group OFFICE\contractors to the Temp Staff access policy rule. The resources that the group can access are dependent on the existing properties of the rule in addition to the site's assignment and entitlement policies.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Set-BrokerAccessPolicyRule 'Temp Staff' -ExcludedClientIPFilterEnabled $true -AddExcludedClientIPs '10.15.0.0/16' -AllowedConnections ViaAG

Modifies the Temp Staff access policy rule to remove access to any user device with an IP address matching 10.15.0.0/16, and requires that all connections by the rule must come through Access Gateway (assuming that the included SmartAccess tags filter is enabled).
Set-BrokerAccessPolicyRuleMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for AccessPolicyRule

Syntax


Detailed Description


Related topics

Parameters

- **-AccessPolicyRuleId** <Int32>

  Specifies the AccessPolicyRule object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByPropertyName)</td>
<td></td>
</tr>
</tbody>
</table>

- **-InputObject** <AccessPolicyRule[]>

  Specifies the AccessPolicyRule objects whose Metadata is to be created/updated.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>-AccessPolicyRuleName</strong></td>
<td>Specifies the AccessPolicyRule object whose Metadata is to be created/updated by name.</td>
</tr>
<tr>
<td><strong>-Name</strong></td>
<td>Specifies the name of the Metadata member to be created/updated</td>
</tr>
<tr>
<td><strong>-Value</strong></td>
<td>Specifies the value of the Metadata member to be created/updated</td>
</tr>
<tr>
<td><strong>-Map</strong></td>
<td>Specifies a hashtable containing name/value pairs to be used to create or update Metadata members</td>
</tr>
<tr>
<td><strong>-PassThru</strong></td>
<td>This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>
- **LoggingId**<Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

  | Required? | false |
  | Default Value | False |
  | Accept Pipeline Input? | false |

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

  | Required? | false |
  | Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
  | Accept Pipeline Input? | false |

**Input Type**


**Return Values**

None or Citrix.BrokerAdmin.SDK.BrokerAccessPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerAccessPolicyRule object.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerAccessPolicyRuleMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"

This command creates/updates the Metadata "MyMetadataName" key-value pair for the AccessPolicyRule whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerAccessPolicyRule | Set-BrokerAccessPolicyRuleMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the AccessPolicyRule in the site

EXAMPLE 3

C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerAccessPolicyRuleMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the AccessPolicyRule in the site whose name is 'objname'
Set-BrokerAppAssignmentPolicyRule

Oct 10, 2013
Modifies an existing application rule in the site's assignment policy.

Syntax


Detailed Description

The Set-BrokerAppAssignmentPolicyRule cmdlet modifies an existing application rule in the site's assignment policy.

An application rule in the assignment policy defines the users who are entitled to a self-service persistent machine assignment from the rule's desktop group; once assigned the machine can run one or more applications published from the group.

Changing an application rule does not alter machine assignments that have already been made by the rule, nor does it affect active sessions to those machines in any way.

Related topics

New-BrokerAppAssignmentPolicyRule

Get-BrokerAppAssignmentPolicyRule

Rename-BrokerAppAssignmentPolicyRule

Remove-BrokerAppAssignmentPolicyRule

Parameters

- **InputObject** <AppAssignmentPolicyRule[]>

The application rule in the assignment policy to be modified.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>
**-Name**<String>

Specifies the name of the application rule in the assignment policy to be modified.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AddExcludedUsers**<User[]>

Adds the specified users to the excluded users filter of the application rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from this rule.

See the ExcludedUsers parameter for more information.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AddIncludedUsers**<User[]>

Adds the specified users to the included users filter of the application rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

See the IncludedUsers parameter for more information.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Description**<String>

Changes the description of the application rule. The text is purely informational for the administrator; it is never visible to
the end user.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Enabled</em>&lt;Boolean&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

Enables or disables the application rule. A disabled rule is ignored when evaluating the site's assignment policy.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>ExcludedUserFilterEnabled</em>&lt;Boolean&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

Enables or disables the excluded users filter. If the filter is disabled then any user entries in the filter are ignored when assignment policy rules are evaluated.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>ExcludedUsers</em>&lt;User[]&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

Changes the excluded users filter of the application rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from the rule.

This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>IncludedUserFilterEnabled</em>&lt;Boolean&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

Enables or disables the included users filter. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a machine assignment by the application rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.
### -IncludedUsers <User[]>

Changes the included users filter of the application rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

If a user appears explicitly in the excluded users filter of the rule, or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

### -RemoveExcludedUsers <User[]>

Removes the specified users from the excluded users filter of the application rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from this rule.

See the ExcludedUsers parameter for more information.

### -RemoveIncludedUsers <User[]>

Removes the specified users from the included users filter of the application rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

See the IncludedUsers parameter for more information.

### -LoggingId <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level
Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.AppAssignmentPolicyRule The application rule within the assignment policy to be modified.

**Return Values**

None or Citrix.BrokerAdmin.SDK.AppAssignmentPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.AppAssignmentPolicyRule object.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerAppAssignmentPolicyRule 'Temp Staff' -AddIncludedUsers office\interns

Adds the user group OFFICE\interns to the Temp Staff application rule in the assignment policy. This grants all members of that user group an entitlement to a machine in the rule's desktop group. The machines can run applications published from the group. The application session properties obtained using the rule are determined by the rule's other properties.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Set-BrokerAppAssignmentPolicyRule 'Temp Staff' -Enabled $false

Disables the Temp Staff application rule in the assignment policy. This prevents further machine assignments being made using this rule until it is re-enabled. However, access to machines already assigned using the rule is not impacted.
Set-BrokerAppEntitlementPolicyRule

Oct 10, 2013
Modifies an existing application rule in the site's entitlement policy.

Syntax


Detailed Description

The Set-BrokerAppEntitlementPolicyRule cmdlet modifies an existing application rule in the site's entitlement policy.

An application rule in the entitlement policy defines the users who are allowed per-session access to a machine to run one or more applications published from the rule's desktop group.

Changing a rule does not affect existing sessions launched using the rule, but if the change removes an entitlement to a machine that was previously granted, users may be unable to reconnect to a disconnected session on that machine.

Related topics

New-BrokerAppEntitlementPolicyRule
Get-BrokerAppEntitlementPolicyRule
Rename-BrokerAppEntitlementPolicyRule
Remove-BrokerAppEntitlementPolicyRule

Parameters

- **-InputObject** <AppEntitlementPolicyRule[]>

  The application rule in the entitlement policy to be modified.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>

  The name of the application rule in the entitlement policy to be modified.

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
Required? | true
---|---
Default Value
Accept Pipeline Input? | true (ByPropertyName)

-**PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

| Required? | false
---|---
| Default Value | False
| Accept Pipeline Input? | false

-**AddExcludedUsers**<User[]>

Adds the specified users to the excluded users filter of the rule, that is, the users and groups who are explicitly denied entitlements to run applications published from the desktop group.

See the ExcludedUsers parameter for more information.

| Required? | false
---|---
| Default Value
| Accept Pipeline Input? | false

-**AddIncludedUsers**<User[]>

Adds the specified users to the included users filter of the rule, that is, the users and groups who are granted an entitlement to an application session by the rule.

See the IncludedUsers parameter for more information.

| Required? | false
---|---
| Default Value
| Accept Pipeline Input? | false

-**Description**<String>

Changes the description of the application rule. The text is purely informational for the administrator, it is never visible to the end user.

| Required? | false
---|---

### -Enabled<Boolean>

Enables or disables the application rule. A disabled rule is ignored when evaluating the site's entitlement policy.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -ExcludedUserFilterEnabled<Boolean>

Enables or disables the excluded users filter. If the filter is disabled then any user entries in the filter are ignored when entitlement policy rules are evaluated.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -ExcludedUsers<User[]>

Changes the excluded users filter of the rule, that is, the users and groups who are explicitly denied entitlements to run applications published from the desktop group.

This can be used to exclude users or groups or users who would otherwise gain access by groups specified in the included users filter.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -IncludedUserFilterEnabled<Boolean>

Enables or disables the included users filter. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to an application session by the application rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

<p>| Required? | false |
| Default Value |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-IncludedUsers</code>&lt;User[]&gt;</td>
<td>Changes the included users filter of the rule, that is, the users and groups who are granted an entitlement to an application session by the rule. If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.</td>
</tr>
<tr>
<td><code>-RemoveExcludedUsers</code>&lt;User[]&gt;</td>
<td>Removes the specified users from the excluded users filter of the application rule, that is, the users and groups who are explicitly denied entitlements to run applications published from the desktop group. See the ExcludedUsers parameter for more information.</td>
</tr>
<tr>
<td><code>-RemoveIncludedUsers</code>&lt;User[]&gt;</td>
<td>Removes the specified users from the included users filter of the rule, that is, the users and groups who are granted an entitlement to an application session by the rule. See the IncludedUsers parameter for more information.</td>
</tr>
<tr>
<td><code>-LoggingId</code>&lt;Guid&gt;</td>
<td>Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.</td>
</tr>
</tbody>
</table>
- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**

Citrix.BrokerAdmin.SDK.AppEntitlementPolicyRule The application rule in the entitlement policy rule to be modified.

**Return Values**

None or Citrix.BrokerAdmin.SDK.AppEntitlementPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.AppEntitlementPolicyRule object.

**Examples**

------------- EXAMPLE 1 ---------------

C:\PS> Set-BrokerAppEntitlementPolicyRule 'Temp Workers' -AddIncludedUsers office\contractors

Adds the user group OFFICE\contractors to those entitled to run applications from the rule's associated desktop group. This grants all members of that group an entitlement to an application session from that group.

------------- EXAMPLE 2 ---------------

C:\PS> Set-BrokerAppEntitlementPolicyRule 'Temp Workers' -Enabled $false

Disables the Temp Workers application rule in the entitlement policy. This prevents further application sessions being launched using this rule until it is re-enabled. However, access to existing application sessions is not affected.
Set-BrokerApplication

Oct 10, 2013
Changes the settings of an application to the value specified in the command.

Syntax


Detailed Description

The Set-BrokerApplication cmdlet changes the value of one or more properties of an application, such as its CpuPriorityLevel or its CommandLineArguments, to the value specified in the command.

This cmdlet lets you change only the settings of the Application object, and not the relationships to other objects. For instance, it does not let you change which users can access this application, or change which desktop groups this application is published to. To do this, you need to remove the existing association, and then add a new association. The following example shows how to change the desktop group that an application is associated with from $group1 to $group2:

Remove-BrokerApplication -DesktopGroup $group1

Add-RemoveApplication -DesktopGroup $group2

You can change properties of both HostedOnDesktop and InstalledOnClient applications but it is not possible to change the ApplicationType. Also, the Name cannot be changed using this cmdlet; to do this, use the Rename-BrokerApplication cmdlet.

Related topics

New-BrokerApplication
Add-BrokerApplication
Get-BrokerApplication
Remove-BrokerApplication
Rename-BrokerApplication

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1348
Parameters

- **InputObject**: `<Application[]>`

  Specifies the application to modify. The Uid of the application can also be substituted for the object.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name**: `<String>`

  Specifies the name of the application to be modified.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **PassThru**: `<SwitchParameter>`

  This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>False</td>
<td>false</td>
</tr>
</tbody>
</table>

- **BrowserName**: `<String>`

  Specifies the name of the application to modify. Note that the BrowserName cannot be changed in this manner; to modify the BrowserName of an application, use the Rename-BrokerApplication cmdlet.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ClientFolder**: `<String>`

  Specifies the folder that the application belongs to as the user sees it. This is the application folder displayed in the Citrix Online Plug-in, in Web Services, and also in the user's start menu. Subdirectories can be specified with '\' character. The following special characters are not allowed: / * ? < > | " :. Note that this property cannot be set for applications of type InstalledOnClient.
-CommandLineArguments<String>

Specifies the command-line arguments to use when launching the executable.

-CommandLineExecutable<String>

Specifies the name of the executable file to launch.

-CpuPriorityLevel<CpuPriorityLevel>

Specifies the CPU priority for the launched executable. Valid values are: Low, BelowNormal, Normal, AboveNormal, and High. Note that this property cannot be set for applications of type InstalledOnClient.

-Description<String>

Specifies the description of the application.

-Enabled<Boolean>

Specifies whether or not this application can be launched.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-IconFromClient**<Boolean>

Specifies if the app icon should be retrieved from the application on the client. This is reserved for possible future use, and all applications of type HostedOnDesktop cannot set or change this value.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-IconUid**<Int32>

Specifies which icon to use for this application. This application is visible both to the administrator (in the consoles) and also to the user. If no icon is specified, then a generic built-in application icon is used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-PublishedName**<String>

Specifies the name seen by end users who have access to this application.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Secure CommandLineArgumentsEnabled**<Boolean>

Specifies whether the command-line arguments should be secured. This is reserved for possible future use, and all applications of type HostedOnDesktop can only have this value set to true.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
- **ShortcutAddedToDesktop**<Boolean>

Specifies whether or not a shortcut to the application should be placed on the user device.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ShortcutAddedToStartMenu**<Boolean>

Specifies whether a shortcut to the application should be placed in the user's start menu on their user device.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **StartMenuFolder**<String>

Specifies the name of the start menu folder that holds the application shortcut (if any). This is valid only for the Citrix Online Plug-in. Subdirectories can be specified with \ character. The following special characters are not allowed: / * ? < > | " :

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **UserFilterEnabled**<Boolean>

Specifies whether the application's user filter is enabled or disabled. Where the user filter is enabled, the application is only visible to users who appear in the filter (either explicitly or by virtue of group membership).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Visible**<Boolean>

Specifies whether or not this application is visible to users. Note that it's possible for an application to be disabled and still visible.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-WaitForPrinterCreation</td>
<td>Specifies whether or not the session waits for the printers to be created before allowing the end-user to interact with the session. Note that this property cannot be set for applications of type InstalledOnClient.</td>
</tr>
<tr>
<td>-WorkingDirectory</td>
<td>Specifies from which working directory the executable is launched from.</td>
</tr>
<tr>
<td>-LoggingId</td>
<td>Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.</td>
</tr>
<tr>
<td>-AdminAddress</td>
<td>Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.</td>
</tr>
</tbody>
</table>

### Table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>-WaitForPrinterCreation</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>-WorkingDirectory</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>-LoggingId</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>-AdminAddress</td>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.Application, or depends on parameter You can pipe the application to be added to Set-
BrokerApplication. You can also pipe some of the other parameters by name.

Return Values

None or Citrix.BrokerAdmin.SDK.Application

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.Application object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerApplication -Name "Notepad" -Description 'Windows Notepad'
Modifies the application that has a Name of "Notepad" so that its description reads Windows Notepad.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> $app = Get-BrokerApplication -BrowserName "Calculator"
C:\PS> Set-BrokerApplication -InputObject $app -Enabled $false
First gets the application with a BrowserName of "Calculator", then modifies that application (by supplying the application object in the first position) so that it is disabled for users.
Set-BrokerApplicationInstanceMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for ApplicationInstance

Syntax

Set-BrokerApplicationInstanceMetadata [-ApplicationInstanceId] <Int64> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerApplicationInstanceMetadata [-ApplicationInstanceId] <Int64> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerApplicationInstanceMetadata [-ApplicationInstanceId] <Int64> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerApplicationInstanceMetadata [-InputObject] <ApplicationInstance[]> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerApplicationInstanceMetadata [-InputObject] <ApplicationInstance[]> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerApplicationInstanceMetadata cmdlet creates/updates metadata key-value pairs for ApplicationInstance. The ApplicationInstance can be specified by InputObject or piping.

Related topics

Parameters

- **-ApplicationInstanceId <Int64>**

Specifies the ApplicationInstance object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-InputObject <ApplicationInstance[]>**

Specifies the ApplicationInstance objects whose Metadata is to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>
-Name <String>

Specifies the name of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Value <String>

Specifies the value of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Map <PSObject>

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-PassThru <SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1356
- **AdminAddress**: <String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

  **Input Type**

  *Citrix.BrokerAdmin.SDK.BrokerApplicationInstance* You can pipe the ApplicationInstance to hold the new or updated metadata.

  **Return Values**

  None or *Citrix.BrokerAdmin.SDK.BrokerApplicationInstance*

  This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a *Citrix.BrokerAdmin.SDK.BrokerApplicationInstance* object.

  **Examples**

  -------------------------- EXAMPLE 1 --------------------------

  C:\PS> Set-BrokerApplicationInstanceMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"
  This command creates/updates the Metadata "MyMetadataName" key-value pair for the ApplicationInstance whose instance is pointed by $obj-Uid

  -------------------------- EXAMPLE 2 --------------------------

  C:\PS> Get-BrokerApplicationInstance | Set-BrokerApplicationInstanceMetadata -Name "MyMetadataName" -Value "1234"
  This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the ApplicationInstance in the site
Set-BrokerApplicationMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for Application

Syntax

Set-BrokerApplicationMetadata [-ApplicationId] <Int32> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerApplicationMetadata [-ApplicationId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerApplicationMetadata [-ApplicationId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerApplicationMetadata [-InputObject] <Application[]> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerApplicationMetadata [-InputObject] <Application[]> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerApplicationMetadata [-ApplicationName] <String> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerApplicationMetadata [-ApplicationName] <String> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerApplicationMetadata cmdlet creates/updates metadata key-value pairs for Application. The Application can be specified by InputObject or piping.

Related topics

Parameters

- **-ApplicationId** <Int32>

  Specifies the Application object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-InputObject** <Application[]>

  Specifies the Application objects whose Metadata is to be created/updated.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>String</td>
<td>Specifies the Application object whose Metadata is to be created/updated by name.</td>
</tr>
<tr>
<td>Map</td>
<td>PSObject</td>
<td>Specifies a hashtable containing name/value pairs to be used to create or update Metadata members</td>
</tr>
<tr>
<td>Name</td>
<td>String</td>
<td>Specifies the name of the Metadata member to be created/updated</td>
</tr>
<tr>
<td>Value</td>
<td>String</td>
<td>Specifies the value of the Metadata member to be created/updated</td>
</tr>
<tr>
<td>PassThru</td>
<td>SwitchParameter</td>
<td>This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.</td>
</tr>
</tbody>
</table>

**Required?**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
<tr>
<td>Map</td>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
<tr>
<td>Name</td>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
<tr>
<td>Value</td>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
<tr>
<td>PassThru</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Default Value**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Value</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Accept Pipeline Input?**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>true (ByValue)</td>
</tr>
<tr>
<td>Map</td>
<td>true (ByValue)</td>
</tr>
<tr>
<td>Name</td>
<td>true (ByPropertyName)</td>
</tr>
<tr>
<td>Value</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId<Guid>**

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerApplication You can pipe the Application to hold the new or updated metadata.

**Return Values**

None or Citrix.BrokerAdmin.SDK.BrokerApplication

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerApplication object.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerApplicationMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"

This command creates/updates the Metadata "MyMetadataName" key-value pair for the Application whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerApplication | Set-BrokerApplicationMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the Application in the site

-------------------------- EXAMPLE 3 --------------------------

C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerApplicationMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the Application in the site whose name is 'objname'
Set-BrokerAssignmentPolicyRule

Oct 10, 2013
Modifies an existing desktop rule in the site's assignment policy.

Syntax


Detailed Description

The Set-BrokerAssignmentPolicyRule cmdlet modifies an existing desktop rule in the site's assignment policy.

A desktop rule in the assignment policy defines the users who are entitled to self-service persistent machine assignments from the rule's desktop group. A rule defines how many machines a user is allowed from the group for delivery of full desktop sessions.

Changing a desktop rule does not alter machine assignments that have already been made by the rule, nor does it affect active sessions to those machines in any way.

Related topics

New-BrokerAssignmentPolicyRule
Get-BrokerAssignmentPolicyRule
Rename-BrokerAssignmentPolicyRule
Remove-BrokerAssignmentPolicyRule

Parameters

-**InputObject** <AssignmentPolicyRule[]>

The desktop rule in the assignment policy to be modified.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | true (ByValue)

-**Name**<String>

Specifies the name of the desktop rule in the assignment policy to be modified.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByPropertyName) |

-**PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

-**AddExcludedUsers**<User[]>

Adds the specified users to the excluded users filter of the rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from this rule.

See the ExcludedUsers parameter for more information.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

-**AddIncludedUsers**<User[]>

Adds the specified users to the included users filter of the rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

See the IncludedUsers parameter for more information.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

-**ColorDepth**<ColorDepth>
Changes the color depth of any desktop sessions to machines assigned by the rule.

Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

| Required? | false |
| Default Value | 
| Accept Pipeline Input? | false |

- **Description**<String>

Changes the description of the desktop rule. The text may be visible to the end user, for example, as a tooltip associated with the desktop entitlement.

A null value indicates that the equivalent setting from the rule's desktop group is used.

| Required? | false |
| Default Value | 
| Accept Pipeline Input? | false |

- **Enabled**<Boolean>

Enables or disables the desktop rule. A disabled rule is ignored when evaluating the site's assignment policy.

| Required? | false |
| Default Value | 
| Accept Pipeline Input? | false |

- **ExcludedUserFilterEnabled**<Boolean>

Enables or disables the excluded users filter. If the filter is disabled then any user entries in the filter are ignored when assignment policy rules are evaluated.

| Required? | false |
| Default Value | 
| Accept Pipeline Input? | false |

- **ExcludedUsers**<User[]>

Changes the excluded users filter of the desktop rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from this rule.
This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-IconUid<Int32>**

Changes the unique ID of the icon used to display the machine assignment entitlement to the user, and of the assigned desktop itself following the assignment.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-IncludedUserFilterEnabled<Boolean>**

Enables or disables the included users filter. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a machine assignment by the rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

For rules that relate to RemotePC desktop groups however, if the included user filter is disabled, the rule is effectively disabled.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-IncludedUsers<User[]>**

Changes the included users filter of the rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

If a user appears explicitly in the excluded users filter of the rule, or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
### -MaxDesktops <Int32>

The number of machines from the rule's desktop group to which a user is entitled. Where an entitlement is granted to a user group rather than an individual, the number of machines applies to each member of the user group independently.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -PublishedName <String>

Changes the published name of the machine assignment entitlement as seen by the user. Changing the published name does not affect the names of desktops that have already been assigned by the rule.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -RemoveExcludedUsers <User[]>

Removes the specified users from the excluded users filter of the rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from this rule.

See the ExcludedUsers parameter for more information.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

### -RemoveIncludedUsers <User[]>

Removes the specified users from the included users filter of the desktop rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

See the IncludedUsers parameter for more information.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |
Accept Pipeline Input? | false
---|---

-SecureIcaRequired\<Boolean>\>

Changes whether the desktop rule requires the SecureICA protocol for desktop sessions to machines assigned using the entitlement.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId\<Guid>\>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress\<String>\>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdmin.SDK.AssignmentPolicyRule The desktop rule within the assignment policy to be modified.

Return Values

None or Citrix.BrokerAdmin.SDK.AssignmentPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.AssignmentPolicyRule object.

Examples
------------- EXAMPLE 1 ------------------

C:\PS> Set-BrokerAssignmentPolicyRule 'Temp Staff' -AddIncludedUsers office\interns
Adds the user group OFFICE\interns to the Temp Staff desktop rule in the assignment policy. This grants all members of that user group entitlements to machines in the rule's desktop group. The number of machine entitlements per user and the session properties of the desktops obtained using the rule are determined by the rule's other properties.

------------- EXAMPLE 2 ------------------

C:\PS> Set-BrokerAssignmentPolicyRule 'Temp Staff' -Enabled $false
Disables the Temp Staff desktop rule in the assignment policy. This prevents further machine assignments being made using this rule until it is re-enabled. However, access to machines already assigned using the rule is not impacted.
Set-BrokerAssignmentPolicyRuleMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for AssignmentPolicyRule

Syntax


Detailed Description

The Set-BrokerAssignmentPolicyRuleMetadata cmdlet creates/updates metadata key-value pairs for AssignmentPolicyRule. The AssignmentPolicyRule can be specified by InputObject or piping.

Related topics

Parameters

- **AssignmentPolicyRuleId** <Int32>

  Specifies the AssignmentPolicyRule object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **InputObject** <AssignmentPolicyRule[]>

  Specifies the AssignmentPolicyRule objects whose Metadata is to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>---</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-AssignmentPolicyRuleName (String)**

Specifies the AssignmentPolicyRule object whose Metadata is to be created/updated by name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Map (PSObject)**

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-Name (String)**

Specifies the name of the Metadata member to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Value (String)**

Specifies the value of the Metadata member to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-PassThru (SwitchParameter)**

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
Default Value | False
--- | ---
Accept Pipeline Input? | false

**-LoggingId** <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress** <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdmin.SDK.BrokerAssignmentPolicyRule You can pipe the AssignmentPolicyRule to hold the new or updated metadata.

Return Values

None or Citrix.BrokerAdmin.SDK.BrokerAssignmentPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerAssignmentPolicyRule object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerAssignmentPolicyRuleMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"
This command creates/updates the Metadata "MyMetadataName" key-value pair for the AssignmentPolicyRule whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerAssignmentPolicyRule | Set-BrokerAssignmentPolicyRuleMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the AssignmentPolicyRule in the site
EXAMPLE 3

C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerAssignmentPolicyRuleMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the AssignmentPolicyRule in the site whose name is 'objname'
Set-BrokerCatalog

Sets the properties of a catalog.

Syntax


Detailed Description

The Set-BrokerCatalog cmdlet sets properties of a catalog or set of catalogs. The catalog can be specified by name, in which case only one catalog can be specified, or one or more catalog instances can be passed to the command either by piping or by using the -InputObject parameter.

Related topics

Get-BrokerCatalog
New-BrokerCatalog
Remove-BrokerCatalog
Rename-BrokerCatalog

Parameters

-InputObject <Catalog[]>

Specifies the catalog objects to modify.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name <String>

Identifies the catalog to modify.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------</td>
</tr>
</tbody>
</table>

**-PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Description**<String>

Supplies the new value of the Description property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-IsRemotePC**<Boolean>

Supplies a new value for IsRemotePC.

IsRemotePC can only be enabled when:

- SessionSupport is SingleSession
- MachinesArePhysical is true.

IsRemotePC can only be set from true to false when no RemotePCAccount references this catalog, and when no Remote PC relationship exists between this catalog and a desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-MinimumFunctionalLevel**<FunctionalLevel>

The new minimum FunctionalLevel required for machines to work successfully in the catalog. If this is higher than the FunctionalLevel of any machines already in the catalog, they will immediately cease to function.

Valid values are L5, L7

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>Type</td>
</tr>
<tr>
<td>-----------</td>
<td>--------</td>
</tr>
<tr>
<td>-ProvisioningSchemeId</td>
<td>Guid</td>
</tr>
<tr>
<td>-PvsAddress</td>
<td>String</td>
</tr>
<tr>
<td>-PvsDomain</td>
<td>String</td>
</tr>
<tr>
<td>-LoggingId</td>
<td>Guid</td>
</tr>
<tr>
<td>-AdminAddress</td>
<td>String</td>
</tr>
</tbody>
</table>
name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.Catalog You can pipe the catalogs to be modified to Set-BrokerCatalog.

**Return Values**

None or Citrix.BrokerAdmin.SDK.Catalog

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.Catalog object.

**Notes**

A catalog's Name property cannot be changed by Set-BrokerCatalog. To rename a catalog use Rename-BrokerCatalog.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerCatalog -Name "MyCatalog" -Description "New Description"
This example specifies a catalog by name and sets its description.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> $permCatalogs = Get-BrokerCatalog -AllocationType Static
C:\PS> Set-BrokerCatalog -InputObject $permCatalogs -Description "Permanently assigned machines"
This example sets the description for all catalogs with AllocationType 'Static'.
Set-BrokerCatalogMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for Catalog

Syntax

Set-BrokerCatalogMetadata [-CatalogId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerCatalogMetadata [-CatalogId] <Int32> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerCatalogMetadata [-CatalogId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerCatalogMetadata [-InputObject] <Catalog[]> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerCatalogMetadata [-InputObject] <Catalog[]> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerCatalogMetadata [-CatalogName] <String> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerCatalogMetadata [-CatalogName] <String> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerCatalogMetadata cmdlet creates/updates metadata key-value pairs for Catalog. The Catalog can be specified by InputObject or piping.

Related topics

Parameters

- **CatalogId**<Int32>
  Specifies the Catalog object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **InputObject**<Catalog[]>
  Specifies the Catalog objects whose Metadata is to be created/updated.

https://docs.citrix.com
<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-CatalogName**: `<String>`

Specifies the Catalog object whose Metadata is to be created/updated by name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Name**: `<String>`

Specifies the name of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Value**: `<String>`

Specifies the value of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Map**: `<PSObject>`

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-PassThru**: `<SwitchParameter>`

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.
**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

---

**Examples**

-------------------- EXAMPLE 1 --------------------

C:\PS> Set-BrokerCatalogMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"
This command creates/updates the Metadata "MyMetadataName" key-value pair for the Catalog whose instance is pointed by $obj-Uid

-------------------- EXAMPLE 2 --------------------

C:\PS> Get-BrokerCatalog | Set-BrokerCatalogMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the Catalog in the site

-------------------------- EXAMPLE 3 --------------------------

C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerCatalogMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the Catalog in the site whose name is 'objname'
Set-BrokerConfigurationSlotMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for ConfigurationSlot

Syntax

Set-BrokerConfigurationSlotMetadata [-ConfigurationSlotId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerConfigurationSlotMetadata [-ConfigurationSlotId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerConfigurationSlotMetadata [-ConfigurationSlotId] <Int32> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerConfigurationSlotMetadata [-InputObject] <ConfigurationSlot[]> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerConfigurationSlotMetadata [-InputObject] <ConfigurationSlot[]> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerConfigurationSlotMetadata [-ConfigurationSlotName] <String> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerConfigurationSlotMetadata [-ConfigurationSlotName] <String> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerConfigurationSlotMetadata cmdlet creates/updates metadata key-value pairs for ConfigurationSlot. The ConfigurationSlot can be specified by InputObject or piping.

Related topics

Parameters

- **-ConfigurationSlotId** <Int32>

Specifies the ConfigurationSlot object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-InputObject** <ConfigurationSlot[]>

Specifies the ConfigurationSlot objects whose Metadata is to be created/updated.
<table>
<thead>
<tr>
<th><strong>-ConfigurationSlotName</strong>&lt;String&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the ConfigurationSlot object whose Metadata is to be created/updated by name.</td>
</tr>
</tbody>
</table>

| **Required?** | true |
| **Default Value** | |
| **Accept Pipeline Input?** | true (ByValue, ByPropertyName) |

<table>
<thead>
<tr>
<th><strong>-Name</strong>&lt;String&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the name of the Metadata member to be created/updated</td>
</tr>
</tbody>
</table>

| **Required?** | true |
| **Default Value** | |
| **Accept Pipeline Input?** | true (ByPropertyName) |

<table>
<thead>
<tr>
<th><strong>-Value</strong>&lt;String&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the value of the Metadata member to be created/updated</td>
</tr>
</tbody>
</table>

| **Required?** | true |
| **Default Value** | |
| **Accept Pipeline Input?** | true (ByPropertyName) |

<table>
<thead>
<tr>
<th><strong>-Map</strong>&lt;PSObject&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies a hashtable containing name/value pairs to be used to create or update Metadata members</td>
</tr>
</tbody>
</table>

| **Required?** | true |
| **Default Value** | |
| **Accept Pipeline Input?** | true (ByValue) |

<table>
<thead>
<tr>
<th><strong>-PassThru</strong>&lt;SwitchParameter&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.</td>
</tr>
</tbody>
</table>
### -LoggingId <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerConfigurationSlot You can pipe the ConfigurationSlot to hold the new or updated metadata.

**Return Values**

None or Citrix.BrokerAdmin.SDK.BrokerConfigurationSlot

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerConfigurationSlot object.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerConfigurationSlotMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"

This command creates/updates the Metadata "MyMetadataName" key-value pair for the ConfigurationSlot whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerConfigurationSlot | Set-BrokerConfigurationSlotMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the ConfigurationSlot in the site.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerConfigurationSlotMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the ConfigurationSlot in the site whose name is 'objname'
Set-BrokerControllerMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for Controller

Syntax

Set-BrokerControllerMetadata [-ControllerId] <Int32> [-Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>]] [<CommonParameters>]

Set-BrokerControllerMetadata [-ControllerId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>]] [<CommonParameters>]

Set-BrokerControllerMetadata [-ControllerId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>]] [<CommonParameters>]

Set-BrokerControllerMetadata [-ControllerId] <Int32> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>]] [<CommonParameters>]


Set-BrokerControllerMetadata [-ControllerName] <String> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>]] [<CommonParameters>]

Set-BrokerControllerMetadata [-ControllerName] <String> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>]] [<CommonParameters>]

Detailed Description

The Set-BrokerControllerMetadata cmdlet creates/updates metadata key-value pairs for Controller. The Controller can be specified by InputObject or piping.

Related topics

Parameters

-ControllerId <Int32>

Specifies the Controller object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-InputObject <Controller[]>

Specifies the Controller objects whose Metadata is to be created/updated.

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1385
## -ControllerName <String>

Specifies the Controller object whose Metadata is to be created/updated by name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

## -Name <String>

Specifies the name of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

## -Value <String>

Specifies the value of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

## -Map <PSObject>

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

## -PassThru <SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId<Guid>**

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### Input Type

Citrix.BrokerAdmin.SDK.BrokerController You can pipe the Controller to hold the new or updated metadata.

### Return Values

None or Citrix.BrokerAdmin.SDK.BrokerController

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerController object.

### Examples

-------------------------- EXAMPLE 1 --------------------------

```
C:\PS> Set-BrokerControllerMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"
```

This command creates/updates the Metadata "MyMetadataName" key-value pair for the Controller whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

```
C:\PS> Get-BrokerController | Set-BrokerControllerMetadata -Name "MyMetadataName" -Value "1234"
```

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1387
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the Controller in the site

-------------------------- EXAMPLE 3 --------------------------

C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerControllerMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the Controller in the site whose name is 'objname'
Set-BrokerDBConnection

Oct 10, 2013
Specifies the Citrix Broker Service's database connection string.

Syntax
Set-BrokerDBConnection [-DBConnection] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-Force]
[-<CommonParameters>]

Detailed Description
Specifies the database connection string for use by the currently selected Citrix Broker Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

It is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to a Broker SDK cmdlet.

Related topics
Get-BrokerServiceStatus
Get-BrokerDBConnection
Test-BrokerDBConnection

Parameters
-DBConnection <String>
Specifies the new database connection string for the currently selected Citrix Broker Service instance.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1389
-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Force**<SwitchParameter>

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

None You cannot pipe input into this cmdlet.

**Return Values**

System.String

The Set-BrokerDBConnection cmdlet returns a string describing the new status of the selected Citrix Broker Service instance. Possible values are:

-- **OK:**

The service instance is configured with a valid database and service schema. The service is operational.

-- **DBUnconfigured:**
No database connection string is set for the service instance.

-- DBRejectedConnection:
The database server rejected the logon from the service instance. This is typically caused by invalid credentials.

-- InvalidDBConfigured:
The specified database does not exist, is not visible to the service instance, or the service's schema within the database is invalid.

-- DBNotFoundException:
The specified database could not be located with the given connection string.

-- DBNewerVersionThanService:
The service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

-- DBOlderVersionThanService:
The service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

-- DBVersionChangeInProgress:
A database schema upgrade is in progress.

-- PendingFailure:
Connectivity between the service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

-- Failed:
Connectivity between the service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

-- Unknown:
Service status cannot be determined.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS>Set-BrokerDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;Trusted_Connection=True"
Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

-------------------------- EXAMPLE 2 --------------------------
C:\PS>Set-BrokerDBConnection -DBConnection $null
Resets the service instance's database connection string. The service instance resets and returns to an idle state until a
valid new database connection string is specified.
Set-BrokerDesktopGroup

Adjusts the settings of a broker desktop group.

Syntax


Detailed Description

The Set-BrokerDesktopGroup cmdlet is used to disable or enable an existing broker desktop group or to alter its settings.

Related topics

Get-BrokerDesktopGroup
New-BrokerDesktopGroup
Rename-BrokerDesktopGroup
Remove-BrokerDesktopGroup

Parameters

-InputObject <DesktopGroup[]>
Specifies the desktop groups to adjust.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-Name**<String>

Specifies the desktop groups to adjust, based on their Name property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AutomaticPowerOnForAssigned**<Boolean>

Specifies whether assigned desktops in the desktop group should be automatically started at the start of peak time periods. Only relevant for groups whose DesktopKind is Private.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AutomaticPowerOnForAssignedDuringPeak**<Boolean>

Specifies whether assigned desktops in the desktop group should be automatically started throughout peak time periods. Only relevant for groups whose DesktopKind is Private and which have AutomaticPowerOnForAssigned set to true.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**-ColorDepth<ColorDepth>-**

Specifies the color depth that the ICA session should use for desktops in this group. Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-DeliveryType<DeliveryType>-**

Specifies whether desktops, applications, or both, can be delivered from machines contained within the desktop group. Desktop groups with a DesktopKind of Private cannot be used to deliver both desktops and applications.

When changing the delivery type to desktops only, there must be no remaining desktop-hosted applications associated with the group, or application-specific assignment/entitlement policy rules for the group.

When changing the delivery type to applications only, there must be no remaining client-hosted applications associated with the group, or desktop-specific assignment/entitlement policy rules for the group.

Valid values are DesktopsOnly, AppsOnly, and DesktopsAndApps.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-Description<String>-**

A description for this desktop group useful for administrators of the site.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-Enabled<Boolean>-**

Whether the desktop group should be in the enabled state; disabled desktop groups do not appear to users.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |
-IconUid<Int32>

The UID of the broker icon to be displayed to users for their desktop(s) in this desktop group.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-InMaintenanceMode<Boolean>

Whether the desktop should be put into maintenance mode; a desktop group in maintenance mode will not allow users to connect or reconnect to their desktops.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-IsRemotePC<Boolean>

Supplies a new value for IsRemotePC.

IsRemotePC can only be enabled when:

- SessionSupport is SingleSession
- DeliveryType is DesktopsOnly
- DesktopKind is Private

IsRemotePC can be switched from true to false only if no RemotePC relationship exists between a catalog and this desktop group.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-MinimumFunctionalLevel<FunctionalLevel>

The new minimum FunctionalLevel required for machines to work successfully in the desktop group. If this is higher than the FunctionalLevel of any machines already in the desktop group, they will immediately cease to function.

| Required? | false |
| Default Value | |

<table>
<thead>
<tr>
<th><strong>-OffPeakBufferSizePercent</strong> &lt;Int32&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>The percentage of machines in the desktop group that should be kept available in an idle state outside peak hours.</td>
</tr>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-OffPeakDisconnectAction</strong> &lt;SessionChangeHostingAction&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>The action to be performed after a configurable period of a user session disconnecting outside peak hours. Possible values are Nothing, Suspend, or Shutdown.</td>
</tr>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-OffPeakDisconnectTimeout</strong> &lt;Int32&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>The number of minutes before the configured action should be performed after a user session disconnects outside peak hours.</td>
</tr>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-OffPeakExtendedDisconnectAction</strong> &lt;SessionChangeHostingAction&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>The action to be performed after a second configurable period of a user session disconnecting outside peak hours. Possible values are Nothing, Suspend, or Shutdown.</td>
</tr>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-OffPeakExtendedDisconnectTimeout</strong> &lt;Int32&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>The number of minutes before the second configured action should be performed after a user session disconnects outside peak hours.</td>
</tr>
<tr>
<td>Required?</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
<tr>
<td><code>-OffPeakLogOffAction</code></td>
</tr>
<tr>
<td><code>-OffPeakLogOffTimeout</code></td>
</tr>
<tr>
<td><code>-PeakBufferSizePercent</code></td>
</tr>
<tr>
<td><code>-PeakDisconnectAction</code></td>
</tr>
<tr>
<td><code>-PeakDisconnectTimeout</code></td>
</tr>
</tbody>
</table>
The number of minutes before the configured action should be performed after a user session disconnects in peak hours.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

**-PeakExtendedDisconnectAction**<SessionChangeHostingAction>

The action to be performed after a second configurable period of a user session disconnecting in peak hours. Possible values are Nothing, Suspend, or Shutdown.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

**-PeakExtendedDisconnectTimeout**<Int32>

The number of minutes before the second configured action should be performed after a user session disconnects in peak hours.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

**-PeakLogOffAction**<SessionChangeHostingAction>

The action to be performed after a configurable period of a user session ending in peak hours. Possible values are Nothing, Suspend, or Shutdown.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

**-PeakLogOffTimeout**<Int32>

The number of minutes before the configured action should be performed after a user session ends in peak hours.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |
-**ProtocolPriority**<String[]>

A list of protocol names in the order in which they should be attempted for use during connection.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**PublishedName**<String>

The name that will be displayed to users for their desktop(s) in this desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**SecureIcaRequired**<Boolean>

Whether HDX connections to desktops in the new desktop group require the use of a secure protocol.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**ShutdownDesktopsAfterUse**<Boolean>

Whether desktops in this desktop group should be automatically shut down when each user session completes (only relevant to power-managed desktops).

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**TimeZone**<String>

The time zone in which this desktop group's machines reside.

The time zone must be specified for any of the group's automatic power management settings to take effect. Automatic power management operations include pool management (power time schemes), reboot schedules, session disconnect and logoff actions, and powering on assigned machines etc.
- **TurnOnAddedMachine**: Boolean

This flag specifies whether the Broker Service should attempt to power on machines when they are added to the desktop group.

- **LoggingId**: Guid

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

- **AdminAddress**: String

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

**Input Type**

Citrix.BrokerAdmin.SDK.DesktopGroup You can pipe the desktop groups to be adjusted to Set-BrokerDesktopGroup.

**Return Values**

None or Citrix.BrokerAdmin.SDK.DesktopGroup

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.DesktopGroup object.
Examples

--------------- EXAMPLE 1 ---------------

C:\PS> Set-BrokerDesktopGroup EMEA* -InMaintenanceMode $true -PassThru
Sets all desktop groups with names starting "EMEA" into maintenance mode, returning the set of desktop groups.

--------------- EXAMPLE 2 ---------------

C:\PS> Get-BrokerDesktopGroup -InMaintenanceMode $true | Set-BrokerDesktopGroup -Enabled $false
Disable all desktop groups that are in maintenance mode.
Set-BrokerDesktopGroupMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for DesktopGroup

Syntax

Set-BrokerDesktopGroupMetadata [-DesktopGroupId] <Int32> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerDesktopGroupMetadata [-DesktopGroupId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerDesktopGroupMetadata [-DesktopGroupId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerDesktopGroupMetadata [-InputObject] <DesktopGroup[]> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerDesktopGroupMetadata [-InputObject] <DesktopGroup[]> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerDesktopGroupMetadata [-DesktopGroupName] <String> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerDesktopGroupMetadata [-DesktopGroupName] <String> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerDesktopGroupMetadata cmdlet creates/updates metadata key-value pairs for DesktopGroup. The DesktopGroup can be specified by InputObject or piping.

Related topics

Parameters

-DesktopGroupId <Int32>

Specifies the DesktopGroup object whose Metadata is to be created/updated by ID.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-InputObject <DesktopGroup[]>

Specifies the DesktopGroup objects whose Metadata is to be created/updated.
-DesktopGroupName:String

Specifies the DesktopGroup object whose Metadata is to be created/updated by name.

-Map:PSObject

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

-Name:String

Specifies the name of the Metadata member to be created/updated

-Value:String

Specifies the value of the Metadata member to be created/updated

-PassThru:SwitchParameter

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.
### -LoggingId (Guid)

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the `Start-LogHighLevelOperation` and `Stop-LogHighLevelOperation` cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress (String)

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td><em>Localhost. Once a value is provided by any cmdlet, this value will become the default.</em></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerDesktopGroup You can pipe the DesktopGroup to hold the new or updated metadata.

**Return Values**

None or Citrix.BrokerAdmin.SDK.BrokerDesktopGroup

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerDesktopGroup object.

**Examples**

-------------  EXAMPLE 1  ----------------

```powershell
C:\PS> Set-BrokerDesktopGroupMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"
```

This command creates/updates the Metadata "MyMetadataName" key-value pair for the DesktopGroup whose instance is pointed by $obj-Uid

-------------  EXAMPLE 2  ----------------

```powershell
C:\PS> Get-BrokerDesktopGroup | Set-BrokerDesktopGroupMetadata -Name "MyMetadataName" -Value "1234"
```
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the DesktopGroup in the site.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerDesktopGroupMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the DesktopGroup in the site whose name is 'objname'.
Set-BrokerEntitlementPolicyRule

Oct 10, 2013
Modifies an existing desktop rule in the site's entitlement policy.

Syntax


Detailed Description

The Set-BrokerEntitlementPolicyRule cmdlet modifies an existing desktop rule in the site's entitlement policy.

A desktop rule in the entitlement policy defines the users who are allowed per-session access to a machine from the rule's associated desktop group to run a full desktop session.

Changing a rule does not affect existing sessions launched using the rule, but if the change removes an entitlement to a machine that was previously granted, users may be unable to reconnect to a disconnected session on that machine.

Related topics

New-BrokerEntitlementPolicyRule
Get-BrokerEntitlementPolicyRule
Rename-BrokerEntitlementPolicyRule
Remove-BrokerEntitlementPolicyRule

Parameters

- **InputObject** <EntitlementPolicyRule>[]

The desktop rule in the entitlement policy to be modified.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>
- **Name**: String

  The name of the desktop rule in the entitlement policy to be modified.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **PassThru**: SwitchParameter

  This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AddExcludedUsers**: User[]

  Adds the specified users to the excluded users filter of the rule, that is, the users and groups who are explicitly denied an entitlement to a desktop session from this rule.

  See the ExcludedUsers parameter for more information.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AddIncludedUsers**: User[]

  Adds the specified users to the included users filter of the rule, that is, the users and groups who are granted an entitlement to a desktop session by the rule.

  See the IncludedUsers parameter for more information.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ColorDepth**: ColorDepth

  Changes the color depth of any desktop sessions launched by a user from this entitlement. Existing sessions are not
affected.

Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

A null value indicates that the equivalent setting from the rule's desktop group is used.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-Description<String>

Changes the description of the desktop rule. The text may be visible to the end user, for example, as a tooltip associated with the desktop entitlement.

A null value indicates that the equivalent setting from the rule's desktop group is used.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-Enabled<Boolean>

Enables or disables the desktop rule. A disabled rule is ignored when evaluating the site's entitlement policy.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-ExcludedUserFilterEnabled<Boolean>

Enables or disables the excluded users filter. If the filter is disabled then any user entries in the filter are ignored when entitlement policy rules are evaluated.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-ExcludedUsers<User>[]

Changes the excluded users filter of the desktop rule, that is, the users and groups who are explicitly denied an entitlement to a desktop session from this rule.
This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-IncludedUserFilterEnabled</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>-IncludedUsers&lt;User[]&gt;</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>-PublishedName&lt;String&gt;</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
Changes the name of the published desktop entitlement as seen by the user.

A null value indicates that the equivalent setting from the rule's desktop group is used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-RemoveExcludedUsers</User[]>-**

Removes the specified users from the excluded users filter of the rule, that is, the users and groups who are explicitly denied an entitlement to a desktop session from this rule.

See the ExcludedUsers parameter for more information.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-RemoveIncludedUsers</User[]>-**

Removes the specified users from the included users filter of the desktop rule, that is, the users and groups who are granted an entitlement to a desktop session by the rule.

See the IncludedUsers parameter for more information.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SecureicaRequired</Boolean>-**

Changes whether the desktop rule requires the SecureICA protocol for desktop sessions launched using the entitlement.

A null value indicates that the equivalent setting from the rule's desktop group is used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId</Guid>-**

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Studio
Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **-AdminAddress** <String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

  | Required? | false |
  | Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
  | Accept Pipeline Input? | false |

**Input Type**

Citrix.BrokerAdmin.SDK.EntitlementPolicyRule The desktop rule in the entitlement policy to be modified.

**Return Values**

None or Citrix.BrokerAdmin.SDK.EntitlementPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.EntitlementPolicyRule object.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerEntitlementPolicyRule 'Temp Workers' -AddIncludedUsers office\contractors

Adds the user group OFFICE\contractors to the Temp Workers desktop rule of the entitlement policy. This grants all members of that group an entitlement to a desktop session in the rule's associated desktop group. The session properties of the desktops obtained using the rule are determined by the rule's other properties.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Set-BrokerEntitlementPolicyRule 'Temp Workers' -Enabled $false

Disables the Temp Workers desktop rule in the entitlement policy. This prevents further desktop sessions being launched using this rule until it is re-enabled. However, access to existing desktop sessions is not affected.
Set-BrokerEntitlementPolicyRuleMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for EntitlementPolicyRule

Syntax


Detailed Description

The Set-BrokerEntitlementPolicyRuleMetadata cmdlet creates/updates metadata key-value pairs for EntitlementPolicyRule. The EntitlementPolicyRule can be specified by InputObject or piping.

Related topics

Parameters

-**EntitlementPolicyRuleId**<Int32>

Specifies the EntitlementPolicyRule object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**InputObject**<EntitlementPolicyRule[]>

Specifies the EntitlementPolicyRule objects whose Metadata is to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
</table>
-EntitlementPolicyRuleName <String>

Specifies the EntitlementPolicyRule object whose Metadata is to be created/updated by name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-Name <String>

Specifies the name of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Value <String>

Specifies the value of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Map <PSObject>

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-PassThru <SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>---------------</td>
<td>-------</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**


**Return Values**

None or Citrix.Broker.Admin.SDK.BrokerEntitlementPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerEntitlementPolicyRule object.

**Examples**

---------- EXAMPLE 1 ----------

C:\PS> Set-BrokerEntitlementPolicyRuleMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"

This command creates/updates the Metadata "MyMetadataName" key-value pair for the EntitlementPolicyRule whose instance is pointed by $obj-Uid

---------- EXAMPLE 2 ----------

C:\PS> Get-BrokerEntitlementPolicyRule | Set-BrokerEntitlementPolicyRuleMetadata -Name "MyMetadataName" -Value "1234"

This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the EntitlementPolicyRule in the site
EXAMPLE 3

C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerEntitlementPolicyRuleMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the EntitlementPolicyRule in the site whose name is 'objname'
Set-BrokerHostingPowerAction

Oct 10, 2013
Changes the priority of one or more pending power actions.

Syntax


Detailed Description

The Set-BrokerHostingPowerAction cmdlet modifies an existing power action in the site's power action queue. The only property of power actions you can change, is the current priority of the action.

For a detailed description of the queuing mechanism, see 'help about_Broker_PowerManagement'.

Related topics

Get-BrokerHostingPowerAction
New-BrokerHostingPowerAction
Remove-BrokerHostingPowerAction

Parameters

-**InputObject** <HostingPowerAction[]>

The power action whose priority is to be changed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**MachineName** <String>

Changes the priority of actions that are for machines whose name (of the form domain\machine) matches the specified string.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
-**PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

-**ActualPriority**<Int32>

Specifies a new priority value for the action in the queue.

This priority is the current action priority; the 'base' or original priority for actions cannot be altered. Numerically lower priority values indicate more important actions that are processed in preference to actions with numerically higher priority settings.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |
Citrix.BrokerAdmin.SDK.HostingPowerAction The power action whose priority is to be changed.

Return Values

None or Citrix.BrokerAdmin.SDK.HostingPowerAction

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.HostingPowerAction object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerHostingPowerAction -MachineName 'XD_VDA1' -ActualPriority 25
Sets the current priority of actions for the machine called 'XD_VDA1' to 25. Numerically lower priority values indicate more important actions that will be processed in preference to actions with numerically higher priority settings.
Set-BrokerHostingPowerActionMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for HostingPowerAction

Syntax

Set-BrokerHostingPowerActionMetadata [-HostingPowerActionId] <Int64> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerHostingPowerActionMetadata [-HostingPowerActionId] <Int64> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerHostingPowerActionMetadata [-HostingPowerActionId] <Int64> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]


Set-BrokerHostingPowerActionMetadata [-HostingPowerActionName] <String> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerHostingPowerActionMetadata [-HostingPowerActionName] <String> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerHostingPowerActionMetadata cmdlet creates/updates metadata key-value pairs for HostingPowerAction. The HostingPowerAction can be specified by InputObject or piping.

Related topics

Parameters

-HostingPowerActionId <Int64>

Specifies the HostingPowerAction object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-InputObject <HostingPowerAction[]>

Specifies the HostingPowerAction objects whose Metadata is to be created/updated.
<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-HostingPowerActionName**<String>

Specifies the HostingPowerAction object whose Metadata is to be created/updated by name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Name**<String>

Specifies the name of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Value**<String>

Specifies the value of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Map**<PSObject>

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.
-LoggingId: <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-AdminAddress: <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

---

### Return Values

None or Citrix.Broker.Admin.SDK.BrokerHostingPowerAction

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerHostingPowerAction object.

### Examples

------------- EXAMPLE 1 -------------

C:\PS> Set-BrokerHostingPowerActionMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"

This command creates/updates the Metadata "MyMetadataName" key-value pair for the HostingPowerAction whose instance is pointed by $obj-Uid

------------- EXAMPLE 2 -------------


C:\PS> Get-BrokerHostingPowerAction | Set-BrokerHostingPowerActionMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the HostingPowerAction in the site

-------------------------- EXAMPLE 3 --------------------------

C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerHostingPowerActionMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the HostingPowerAction in the site whose name is 'objname'
Set-BrokerHypervisorAlertMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for HypervisorAlert

Syntax

Set-BrokerHypervisorAlertMetadata [-HypervisorAlertId] <Int64> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerHypervisorAlertMetadata [-HypervisorAlertId] <Int64> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerHypervisorAlertMetadata [-HypervisorAlertId] <Int64> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerHypervisorAlertMetadata [-InputObject] <HypervisorAlert[]> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerHypervisorAlertMetadata [-InputObject] <HypervisorAlert[]> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerHypervisorAlertMetadata cmdlet creates/updates metadata key-value pairs for HypervisorAlert. The HypervisorAlert can be specified by InputObject or piping.

Related topics

Parameters

-HypervisorAlertId <Int64>

Specifies the HypervisorAlert object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-InputObject <HypervisorAlert[]>

Specifies the HypervisorAlert objects whose Metadata is to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>
- **Map**<PSObject>
  Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name**<String>
  Specifies the name of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **Value**<String>
  Specifies the value of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **PassThru**<SwitchParameter>
  This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
- **AdminAddress**: `<String>`

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerHypervisorAlert You can pipe the HypervisorAlert to hold the new or updated metadata.

**Return Values**

None or Citrix.BrokerAdmin.SDK.BrokerHypervisorAlert

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerHypervisorAlert object.

**Examples**

-------------- EXAMPLE 1 ---------------

C:\PS> Set-BrokerHypervisorAlertMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"
This command creates/updates the Metadata "MyMetadataName" key-value pair for the HypervisorAlert whose instance is pointed by $obj-Uid

-------------- EXAMPLE 2 ---------------

C:\PS> Get-BrokerHypervisorAlert | Set-BrokerHypervisorAlertMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the HypervisorAlert in the site
Set-BrokerHypervisorConnection

Oct 10, 2013
Sets the properties of a hypervisor connection.

Syntax

Set-BrokerHypervisorConnection [-InputObject] <HypervisorConnection[]> [-PassThru] [-PreferredController <String>] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters>

Set-BrokerHypervisorConnection [-Name] <String> [-PassThru] [-PreferredController <String>] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters>

Detailed Description

The Set-BrokerHypervisorConnection cmdlet sets the properties on a hypervisor connection.

Related topics

Get-BrokerHypervisorConnection
New-BrokerHypervisorConnection
Remove-BrokerHypervisorConnection

Parameters

-**InputObject**<HypervisorConnection[]>

Specifies the hypervisor connection object to adjust.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-**Name**<String>

Specifies the name of the hypervisor connection object to adjust.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-**PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

-**PreferredController**<String>

-**AdminAddress**<String>
-PreferredController<String>

Supplies the new value of the PreferredController property.

-LoggingId<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

Input Type

Citrix Broker Admin SDK HypervisorConnection You can pipe the hypervisor connection to be modified to Set-BrokerHypervisorConnection.

Return Values

None or Citrix Broker Admin SDK HypervisorConnection

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix Broker Admin SDK HypervisorConnection object.
Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> $hvConn = Get-BrokerHypervisorConnection -PreferredController "oldController" -Name "Xen Server Connection"
c:\PS> Set-BrokerHypervisorConnection -InputObject $hvConn -PreferredController "newController"

Updates the preferred controller for a hypervisor connection.
Set-BrokerHypervisorConnectionMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for HypervisorConnection

Syntax

Set-BrokerHypervisorConnectionMetadata [-HypervisorConnectionId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerHypervisorConnectionMetadata [-HypervisorConnectionId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerHypervisorConnectionMetadata [-InputObject] <HypervisorConnection[]> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerHypervisorConnectionMetadata [-InputObject] <HypervisorConnection[]> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerHypervisorConnectionMetadata [-HypervisorConnectionName] <String> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerHypervisorConnectionMetadata [-HypervisorConnectionName] <String> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerHypervisorConnectionMetadata cmdlet creates/updates metadata key-value pairs for HypervisorConnection. The HypervisorConnection can be specified by InputObject or piping.

Related topics

Parameters

-HypervisorConnectionId <Int32>

Specifies the HypervisorConnection object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Input Object <HypervisorConnection[]>

Specifies the HypervisorConnection objects whose Metadata is to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>--</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **HypervisorConnectionName**:<String>

Specifies the HypervisorConnection object whose Metadata is to be created/updated by name.

| Required? | true |
| Default Value |  |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

- **Name**:<String>

Specifies the name of the Metadata member to be created/updated

| Required? | true |
| Default Value |  |
| Accept Pipeline Input? | true (ByPropertyName) |

- **Value**:<String>

Specifies the value of the Metadata member to be created/updated

| Required? | true |
| Default Value |  |
| Accept Pipeline Input? | true (ByPropertyName) |

- **Map**:<PSObject>

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

| Required? | true |
| Default Value |  |
| Accept Pipeline Input? | true (ByValue) |

- **PassThru**:<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? |  |
### -LoggingId <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

#### Input Type

Citrix.BrokerAdmin.SDK.BrokerHypervisorConnection You can pipe the HypervisorConnection to hold the new or updated metadata.

#### Return Values

None or Citrix.BrokerAdmin.SDK.BrokerHypervisorConnection

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerHypervisorConnection object.

#### Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerHypervisorConnectionMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"

This command creates/updates the Metadata "MyMetadataName" key-value pair for the HypervisorConnection whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerHypervisorConnection | Set-BrokerHypervisorConnectionMetadata -Name "MyMetadataName" -Value "1234"

This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the HypervisorConnection in the site
EXAMPLE 3

C:\PS> @(
'name1' = 'value1';
'name2' = 'value2'
) | Set-BrokerHypervisorConnectionMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the HypervisorConnection in the site whose name is 'objname'
Set-BrokerIconMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for Icon

Syntax

Set-BrokerIconMetadata [-IconId] <Int32> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerIconMetadata [-IconId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerIconMetadata [-IconId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerIconMetadata [-InputObject] <Icon[]> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerIconMetadata [-InputObject] <Icon[]> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerIconMetadata cmdlet creates/updates metadata key-value pairs for Icon. The Icon can be specified by InputObject or piping.

Related topics

Parameters

- **-IconId** <Int32>

Specifies the Icon object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-InputObject** <Icon[]>

Specifies the Icon objects whose Metadata is to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>
-Map<PSObject>

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name<String>

Specifies the name of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Value<String>

Specifies the value of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-PassThru<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | false
---|---

- **AdminAddress**: <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**  
Citrix.BrokerAdmin.SDK.BrokerIcon You can pipe the Icon to hold the new or updated metadata.

**Return Values**  
None or Citrix.BrokerAdmin.SDK.BrokerIcon

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerIcon object.

**Examples**

----------- EXAMPLE 1 -----------

C:\PS> Set-BrokerIconMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"
This command creates/updates the Metadata "MyMetadataName" key-value pair for the Icon whose instance is pointed by $obj-Uid

----------- EXAMPLE 2 -----------

C:\PS> Get-BrokerIcon | Set-BrokerIconMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the Icon in the site
Set-BrokerMachine

Sets properties on a machine.

**Syntax**

```
[-HostedMachineId <String>] [-HypervisorConnectionUid <Int32>] [-InMaintenanceMode <Boolean>] [-LoggingId <Guid>]
[AdminAddress <String>] [CommonParameters]

[-HostedMachineId <String>] [-HypervisorConnectionUid <Int32>] [-InMaintenanceMode <Boolean>] [-LoggingId <Guid>]
[AdminAddress <String>] [CommonParameters]
```

**Detailed Description**

The Set-BrokerMachine cmdlet sets properties on a machine or set of machines. You can specify a single machine by name or multiple machine instances can be passed to the command by piping or using the -InputObject parameter.

**Related topics**

- Get-BrokerMachine
- New-BrokerMachine

**Parameters**

- **-InputObject** <Machine[]>

  The machine instances whose properties you want to set.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-MachineName** <String>

  The machine whose properties you want to set.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-PassThru** <SwitchParameter>

  This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected
record.

<table>
<thead>
<tr>
<th>Field</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>AssignedClientName &lt;String&gt;</td>
<td>false</td>
<td>False</td>
<td>false</td>
</tr>
</tbody>
</table>

Changes the client name assignment of the machine. Set this to $null to remove the assignment. You can assign machines to multiple users, a single client IP address, or a single client name, but only to one of these categories at a time.

<table>
<thead>
<tr>
<th>Field</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>AssignedIPAddress &lt;String&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

Changes the client IP address assignment of the machine. Set this to $null to remove the assignment. You can assign machines to multiple users, a single client IP address, or a single client name, but only to one of these categories at a time.

<table>
<thead>
<tr>
<th>Field</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>HostedMachineId &lt;String&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

The unique ID by which the hypervisor recognizes the machine. This may only be set for VMs which are not provisioned by MCS.

<table>
<thead>
<tr>
<th>Field</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>HypervisorConnectionUid &lt;Int32&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

The hypervisor connection that runs the machine. This may only be set for VMs which are not provisioned by MCS.
- **InMaintenanceMode**: Boolean
  Sets whether the machine is in maintenance mode or not. A machine in maintenance mode is not available for new sessions, and for managed machines all automatic power management is disabled.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **LoggingId**: Guid
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **AdminAddress**: String
  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**
Citrix.Broker.Admin.SDK.Machine You can pipe in the machines whose properties you want to set.

**Return Values**
None or Citrix.Broker.Admin.SDK.Machine

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.Machine object.

**Examples**

```
C:\PS> Set-BrokerMachine -MachineName 'MyDomain\MyMachine' -HypervisorConnectionUid 3
```
This example specifies a machine by name and sets its hypervisor connection.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> $machines = Get-BrokerMachine -MachineName 'MyDomain\*'  
C:\PS> Set-BrokerMachine -InputObject $machines -HypervisorConnectionUid 3  
This example finds all machines in domain MyDomain and sets their hypervisor connections.

-------------------------- EXAMPLE 3 --------------------------

This example also finds all machines in domain MyDomain and sets their hypervisor connections.
Set-BrokerMachineCatalog

Oct 10, 2013
Moves one or more machines into a different catalog.

Syntax


Detailed Description

The Set-BrokerMachineCatalog cmdlet is used to move machines into a different catalog. The following properties of the destination catalog must exactly match those of the machine's current catalog otherwise the command fails:

- AllocationType
- ProvisioningType
- PersistUserChanges
- SessionSupport
- IsRemotePC
- MinimumFunctionalLevel
- PhysicalMachines

Changing a machine's catalog does not change the machine's desktop group membership. There is no effect on user sessions present on a machine if its catalog is changed.

Related topics

Set-BrokerMachine
New-BrokerCatalog

Parameters

- **-InputObject** <Machine[]>

  The machine instances that are being moved into a different catalog.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-CatalogUid** <Int32>

  The unique identifier of the catalog into which the machines are being moved.
### -LoggingId<<Guid>>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.Machine You can pipe in the machines that are to be moved into a new catalog.

**Return Values**

None

**Examples**

```
-------------------------- EXAMPLE 1 --------------------------
C:\PS> $catalog = Get-BrokerCatalog MarketingMachines
C:\PS> $machines = Get-BrokerMachine -MachineName 'Marketing\*' # You can specify the domain name
C:\PS> Set-BrokerMachineCatalog $machines -CatalogUid $cat.Uid
```

This example finds all machines in domain Marketing and moves them into a catalog called MarketingMachines.
Set-BrokerMachineCommandMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for MachineCommand

Syntax
Set-BrokerMachineCommandMetadata [-MachineCommandId] <Int64> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]
Set-BrokerMachineCommandMetadata [-MachineCommandId] <Int64> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]
Set-BrokerMachineCommandMetadata [-MachineCommandId] <Int64> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description
The Set-BrokerMachineCommandMetadata cmdlet creates/updates metadata key-value pairs for MachineCommand. The MachineCommand can be specified by InputObject or piping.

Related topics
Parameters
-MachineCommandId <Int64>
Specifies the MachineCommand object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-InputObject <MachineCommand[]>
Specifies the MachineCommand objects whose Metadata is to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>
- **Map** `<PSObject>`

   Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name** `<String>`

   Specifies the name of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **Value** `<String>`

   Specifies the value of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **PassThru** `<SwitchParameter>`

   This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId** `<Guid>`

   Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

---

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1444
- **AdminAddress**: String

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerMachineCommand You can pipe the MachineCommand to hold the new or updated metadata.

**Return Values**

None or Citrix.BrokerAdmin.SDK.BrokerMachineCommand

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerMachineCommand object.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerMachineCommandMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"

This command creates/updates the Metadata "MyMetadataName" key-value pair for the MachineCommand whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerMachineCommand | Set-BrokerMachineCommandMetadata -Name "MyMetadataName" -Value "1234"

This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the MachineCommand in the site
Set-BrokerMachineConfiguration

Oct 10, 2013
Sets the properties of a machine configuration.

Syntax


Detailed Description

Sets the properties of a machine configuration. The encoded settings data must only contain settings that match the SettingsGroup of the associated configuration slot. Use the SDK snap-in that matches the SettingsGroup of the associated configuration slot to generate new encoded settings data or modify existing settings values.

Related topics

New-BrokerMachineConfiguration
Get-BrokerMachineConfiguration
Rename-BrokerMachineConfiguration
Remove-BrokerMachineConfiguration
Add-BrokerMachineConfiguration

Parameters

- **InputObject** <MachineConfiguration[]>

  Machine configuration to modify.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

- **Name** <String>

  Name of machine configuration to modify.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

  | Accept Pipeline Input? | true (ByPropertyName) |
-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

-Description

New description for the machine configuration.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-Policy

New binary array of encoded settings data.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AdminAddress

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.

Accept Pipeline Input? | false

Input Type


Return Values

None or Citrix.BrokerAdmin.SDK.MachineConfiguration

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.MachineConfiguration object.

Examples

-------------------------- EXAMPLE 1 --------------------------

Set-BrokerMachineConfiguration -Name "UPM\Finance Department" -Policy $newPolicy
Use the encoded settings binary data in $newPolicy to update the machine configuration.

-------------------------- EXAMPLE 2 --------------------------

Get-BrokerMachineConfiguration -Name "UPM\*" | Set-BrokerMachineConfiguration -Description "User Profile Management"
Assign the description "User Profile Management" to every machine configuration associated with the UPM slot.
Set-BrokerMachineConfigurationMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for MachineConfiguration

Syntax

Set-BrokerMachineConfigurationMetadata [-MachineConfigurationId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerMachineConfigurationMetadata [-MachineConfigurationId] <Int32> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerMachineConfigurationMetadata [-MachineConfigurationId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]


Set-BrokerMachineConfigurationMetadata [-MachineConfigurationName] <String> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerMachineConfigurationMetadata cmdlet creates/updates metadata key-value pairs for MachineConfiguration. The MachineConfiguration can be specified by InputObject or piping.

Related topics

Parameters

-MachineConfigurationId <Int32>

Specifies the MachineConfiguration object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Input Object <MachineConfiguration[]>

Specifies the MachineConfiguration objects whose Metadata is to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1449
<table>
<thead>
<tr>
<th>Default Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-MachineConfigurationName&lt;String&gt;

Specifies the MachineConfiguration object whose Metadata is to be created/updated by name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-Name&lt;String&gt;

Specifies the name of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByProperty Name)</td>
</tr>
</tbody>
</table>

-Value&lt;String&gt;

Specifies the value of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Map&lt;PSObject&gt;

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-PassThru&lt;SwitchParameter&gt;

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
### -LoggingId <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerMachineConfiguration You can pipe the MachineConfiguration to hold the new or updated metadata.

**Return Values**

None or Citrix.BrokerAdmin.SDK.BrokerMachineConfiguration

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerMachineConfiguration object.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerMachineConfigurationMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"

This command creates/updates the Metadata "MyMetadataName" key-value pair for the MachineConfiguration whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerMachineConfiguration | Set-BrokerMachineConfigurationMetadata -Name "MyMetadataName" -Value "1234"

This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the MachineConfiguration in the site
C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerMachineConfigurationMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the MachineConfiguration in the site whose name is 'objname'
Set-BrokerMachineMaintenanceMode

Oct 10, 2013
Sets whether the specified machine(s) are in maintenance mode.

Syntax


Detailed Description

The cmdlet can be used to set whether a machine is in maintenance mode or not. A machine in maintenance mode is not available for new sessions, and for managed machines all automatic power management is disabled.

There are times when it is necessary to disable desktops. You can do this by setting the InMaintenanceMode property of a desktop to $true. This puts it into maintenance mode. The broker excludes single-session desktops in maintenance mode from brokering decisions and does not start new sessions on them. Existing sessions are unaffected. For multi-session desktops in maintenance mode, reconnections to existing sessions are allowed, but no new sessions are created on the machine.

Desktops in maintenance mode are also excluded from automatic power management, although explicit power actions are still performed.

This cmdlet is equivalent to using the Set-BrokerMachine cmdlet to set the value of only the InMaintenanceMode property.

Related topics
Set-BrokerMachine

Parameters

- **-InputObject** <Machine[]>

The machine instances whose InMaintenanceMode property you want to set.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

- **-MaintenanceMode**<Boolean>

Sets whether the machine is in maintenance mode or not. A machine in maintenance mode is not available for new sessions, and for managed machines all automatic power management is disabled.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | true (ByValue)
---|---

**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

Input Type

Citrix.BrokerAdmin.SDK.Machine You can pipe in the machines whose properties you want to set.

Return Values

None

Examples

----------------------------- EXAMPLE 1 -----------------------------

C:\PS> $machines = Get-BrokerMachine -MachineName 'MyDomain\*'
C:\PS> Set-BrokerMachineMaintenanceMode -InputObject $machines $false

This example finds all machines in domain MyDomain and removes them from maintenance mode by setting their InMaintenanceMode property to false.
Set-BrokerMachineMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for Machine

Syntax

Set-BrokerMachineMetadata [-MachineId] <Int32> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Set-BrokerMachineMetadata [-MachineId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Set-BrokerMachineMetadata [-MachineId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]


Set-BrokerMachineMetadata [-MachineName] <String> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Set-BrokerMachineMetadata [-MachineName] <String> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Set-BrokerMachineMetadata cmdlet creates/updates metadata key-value pairs for Machine. The Machine can be specified by InputObject or piping.

Related topics

Parameters

-MachineId <Int32>

Specifies the Machine object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-InputObject <Machine[]>

Specifies the Machine objects whose Metadata is to be created/updated.
**-MachineName**<String>

Specifies the Machine object whose Metadata is to be created/updated by name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Map**<PSObject>

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-Name**<String>

Specifies the name of the Metadata member to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Value**<String>

Specifies the value of the Metadata member to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.
- **LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.BrokerAdmin.SDK.BrokerMachine You can pipe the Machine to hold the new or updated metadata.

Return Values

None or Citrix.BrokerAdmin.SDK.BrokerMachine

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerMachine object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerMachineMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"
This command creates/updates the Metadata "MyMetadataName" key-value pair for the Machine whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerMachine | Set-BrokerMachineMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the Machine in the site

-------------------------- EXAMPLE 3 --------------------------

C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerMachineMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the Machine in the site whose name is 'objname'
Set-BrokerPowerTimeScheme

Oct 10, 2013
Modifies an existing power time scheme.

Syntax


Detailed Description
The Set-BrokerPowerTimeScheme cmdlet modifies an existing time scheme.

Each power time scheme is associated with a particular desktop group, and covers one or more days of the week, defining which hours of those days are considered peak times and which are off-peak times. In addition, the time scheme defines a pool size value for each hour of the day for the days of the week covered by the time scheme. No one desktop group can be associated with two or more time schemes that cover the same day of the week.

For more information about the power policy mechanism and pool size management, see 'help about_Broker_PowerManagement'.

Related topics
Get-BrokerPowerTimeScheme
Rename-BrokerPowerTimeScheme
New-BrokerPowerTimeScheme
Remove-BrokerPowerTimeScheme

Parameters

- **-InputObject** <PowerTimeScheme[]>
The power time scheme to be changed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

- **-Name** <String>
Identifies the power time scheme to be changed by name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
<tr>
<td><strong>-PassThru</strong>&lt;SwitchParameter&gt;</td>
<td>This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-DaysOfWeek</strong>&lt;TimeSchemeDays&gt;</td>
<td>Changes the pattern of days of the week that the time scheme applies to. The pattern of days is specified as a single value or a list of values, where each value refers to either a single day or defined group of days. Valid values are: Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Weekend and Weekdays.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-DisplayName</strong>&lt;String&gt;</td>
<td>Changes the name that is used by the DesktopStudio console when showing the time scheme.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-PeakHours</strong>&lt;Boolean[]&gt;</td>
<td>Changes the pattern of hours considered 'peak' vs 'off-peak' for the days covered by the time scheme. A 24-entry array of boolean truth values is expected, where the zeroth entry of the array relates to the time period between midnight and 0:59, the first relates to 1am to 1:59 and so on, with the last array element relating to 11 PM to 11:59. If the flag value is $true it means the associated hour of the day is considered a peak time; if $false it means that it is considered off-peak. If fewer than 24 values are supplied, the final missing values are assumed to be 'false', and if more than 24 values are supplied, only the first 24 are used.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td><strong>-PoolSize</strong>&lt;Int32[]&gt;</td>
<td></td>
</tr>
</tbody>
</table>
Changes the requested pool size of running machines at the various hours of the day for the days covered by the time scheme. A 24-entry array of integer values is expected, where the zeroth entry of the array relates to the time period between midnight and 0:59, the first relates to 1am to 1:59 and so on, with the last array element relating to 11 PM to 11:59. The pool size array entry values are either absolute numbers of machines that should be running or are a percentage of the machines in the desktop group that should be running during the associated hour of the day. A value of -1 in the array signifies that no management of the number of running machines should be attempted during the associated hour of the day.

If fewer than 24 values are supplied, the final missing values are assumed to be -1, and if more than 24 values are supplied, only the first 24 are used.

<table>
<thead>
<tr>
<th>Field</th>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>-PoolUsingPercentage&lt;Boolean&gt;</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

Changes whether pool size values from the 'PoolSize' array are evaluated as absolute numbers of running machines or as a percentage of machines in the desktop group that are to be maintained as running.

A value of $true indicates that the pool size array values are percentages of total machines in the desktop group and a value of $false indicates that the pool size array values are absolute numbers of machines to maintain as running.

<table>
<thead>
<tr>
<th>Field</th>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>-LoggingId&lt;Guid&gt;</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Field</th>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>-AdminAddress&lt;String&gt;</td>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>-AdminAddress&lt;String&gt;</td>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
</tbody>
</table>
**Input Type**

Citrix.BrokerAdmin.SDK.PowerTimeScheme The power time scheme to be changed.

**Return Values**

None or Citrix.BrokerAdmin.SDK.PowerTimeScheme

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.PowerTimeScheme object.

**Examples**

------------- EXAMPLE 1 -------------

```
C:\PS> Set-BrokerPowerTimeScheme -Name 'Development Weekdays' -PoolSize ( 0..23 | %{ if ($_ -lt 8 -or $_ -gt 19) { 5 } else { 20 } } )
```

Sets the pool size for the power time scheme named 'Development Weekdays' to be 20 for the time between 8am to 7:59pm, and 5 for other times.
Set-BrokerPowerTimeSchemeMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for PowerTimeScheme

Syntax
Set-BrokerPowerTimeSchemeMetadata [-PowerTimeSchemeId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerPowerTimeSchemeMetadata [-PowerTimeSchemeId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerPowerTimeSchemeMetadata [-PowerTimeSchemeId] <Int32> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerPowerTimeSchemeMetadata [-InputObject] <PowerTimeScheme[]> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerPowerTimeSchemeMetadata [-InputObject] <PowerTimeScheme[]> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerPowerTimeSchemeMetadata [-PowerTimeSchemeName] <String> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerPowerTimeSchemeMetadata [-PowerTimeSchemeName] <String> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
The Set-BrokerPowerTimeSchemeMetadata cmdlet creates/updates metadata key-value pairs for PowerTimeScheme. The PowerTimeScheme can be specified by InputObject or piping.

Related topics
Parameters

- **PowerTimeSchemeId**<Int32>

  Specifies the PowerTimeScheme object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **InputObject**<PowerTimeScheme[]>

  Specifies the PowerTimeScheme objects whose Metadata is to be created/updated.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-PowerTimeSchemeName</td>
<td>Specifies the PowerTimeScheme object whose Metadata is to be created/updated by name.</td>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
<tr>
<td>-Name</td>
<td>Specifies the name of the Metadata member to be created/updated</td>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
<tr>
<td>-Value</td>
<td>Specifies the value of the Metadata member to be created/updated</td>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
<tr>
<td>-Map</td>
<td>Specifies a hashtable containing name/value pairs to be used to create or update Metadata members</td>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
<tr>
<td>-PassThru</td>
<td>This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Set-BrokerPowerTimeSchemeMetadata

**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerPowerTimeScheme

You can pipe the PowerTimeScheme to hold the new or updated metadata.

**Return Values**

None or Citrix.BrokerAdmin.SDK.BrokerPowerTimeScheme

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerPowerTimeScheme object.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerPowerTimeSchemeMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"

This command creates/updates the Metadata "MyMetadataName" key-value pair for the PowerTimeScheme whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------


C:\PS> Get-BrokerPowerTimeScheme | Set-BrokerPowerTimeSchemeMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the PowerTimeScheme in the site

-------------------------- EXAMPLE 3 --------------------------

C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerPowerTimeSchemeMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the PowerTimeScheme in the site whose name is 'objname'
Set-BrokerPrivateDesktop

Change the settings of a private desktop.

Syntax


Detailed Description

Private desktops are automatically created when a machine is added to a desktop group with a DesktopKind of 'Private', and these inherit default properties. Use Set-BrokerPrivateDesktop to change the configuration settings of an existing private desktop.

To specify private desktops, you can choose whether to update by machine name, or by passing a PrivateDesktop or an array of PrivateDesktop objects. You can also use the Uid or an array of Uids instead.

You cannot modify many properties of a private desktop as these contain status information; for example DNSName, RegistrationState, and OSVersion.

Use Add- and Remove- cmdlets to update relationships between private desktops and other objects. For example, you can add a tag to a private desktop with:

Add-BrokerTag $tag -Desktop $desktop.Uid

Similarly, assign users to private desktops with:

Add-BrokerUser $user -PrivateDesktop $desktop

Many of the fields that can be set with this cmdlet can also be set with Set-BrokerMachine, such as MaintenanceMode. Using Set-BrokerMachine is preferred in these cases.

For more information about desktops, see about_Broker_Desktops; for more information about machines, see about_Broker_Machines.

Related topics

Get-BrokerMachine

Parameters

-InputObject <PrivateDesktop[]>
Specifies the desktop or array of desktops to modify. You can also use an integer Uid of the desktop instead.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-MachineName**<String>

Specifies the desktop to modify using its machine name (in the form 'domain\machine').

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AssignedClientName**<String>

Changes the client name assignment of the desktop. Set this to $null to remove the assignment. Desktops can be assigned to multiple users, a single IP address, or a single client name, but only to one of these categories at one time.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AssignedIPAddress**<String>

Changes the IP address assignment of the desktop. Set this to $null to remove the assignment. Desktops can be assigned to multiple users, a single IP address, or a single client name, but only to one of these categories at one time.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
-**ColorDepth**<ColorDepth>

Changes the color depth connections to this desktop should use.

Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit. A value of $null results in the desktop group value being used instead.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Description**<String>

Changes the description of the desktop. This is seen only by Citrix Administrators and is not visible to users.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**IconUid**<Int32>

Changes the icon displayed for this desktop. When this setting is $null, the icon displayed is determined by the desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**InMaintenanceMode**<Boolean>

Changes the maintenance mode setting of a desktop. When a desktop is in maintenance mode, it is not included as a candidate when brokering new sessions, and it does not participate in automatic power management (idle pool); however, it still responds to explicit power operations.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**PublishedName**<String>

Changes the name displayed to the user for this desktop. When this setting is $null, the name displayed is determined by
the PublishedName of the desktop group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SecureIcaRequired**<Boolean>

Changes whether or not SecureICA is required for connections to this desktop. When this setting is $null, the SecureIcaRequired setting from the desktop group is used.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.PrivateDesktop You can pipe PrivateDesktop objects into this cmdlet instead of on the command line with the -InputObject parameter.

**Return Values**

None or Citrix.BrokerAdmin.SDK.PrivateDesktop
This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.PrivateDesktop object.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Set-BrokerPrivateDesktop DOMAIN\Machine1 -ColorDepth SixteenBit
Change the color depth of Machine1 to be 16-bit.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-BrokerPrivateDesktop -InMaintenanceMode $true | Set-BrokerPrivateDesktop -InMaintenanceMode $false
Bring all private desktops currently in maintenance mode back into normal service.
Set-BrokerRebootCycleMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for RebootCycle

Syntax

Set-BrokerRebootCycleMetadata [-RebootCycleId] <Int64> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerRebootCycleMetadata [-RebootCycleId] <Int64> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerRebootCycleMetadata [-RebootCycleId] <Int64> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerRebootCycleMetadata [-InputObject] <RebootCycle[]> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerRebootCycleMetadata [-InputObject] <RebootCycle[]> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerRebootCycleMetadata cmdlet creates/updates metadata key-value pairs for RebootCycle. The RebootCycle can be specified by InputObject or piping.

Related topics

Parameters

-RebootCycleId <Int64>

Specifies the RebootCycle object whose Metadata is to be created/updated by ID.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-InputObject <RebootCycle[]>

Specifies the RebootCycle objects whose Metadata is to be created/updated.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |
-**Name**<String>

Specifies the name of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**Value**<String>

Specifies the value of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**Map**<PSObject>

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerRebootCycle You can pipe the RebootCycle to hold the new or updated metadata.

**Return Values**

None or Citrix.BrokerAdmin.SDK.BrokerRebootCycle

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerRebootCycle object.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerRebootCycleMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"
This command creates/updates the Metadata "MyMetadataName" key-value pair for the RebootCycle whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerRebootCycle | Set-BrokerRebootCycleMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the RebootCycle in the site
Set-BrokerRebootSchedule

Updates the values of one or more desktop group reboot schedules.

Syntax


Detailed Description

The Set-BrokerRebootSchedule cmdlet is used to alter the settings of an existing desktop group reboot schedule.

Related topics

Get-BrokerRebootSchedule
New-BrokerRebootSchedule
Remove-BrokerRebootSchedule

Parameters

- **-InputObject** <RebootSchedule[]>

  The reboot schedule to be modified.

  **Required?**
  true

  **Default Value**
  

  **Accept Pipeline Input?**
  true (ByValue)

- **-DesktopGroupName** <String>

  The name of the desktop group whose reboot schedule is to be modified.

  **Required?**
  true

  **Default Value**
  

  **Accept Pipeline Input?**
  true (ByPropertyName)

- **-PassThru**<SwitchParameter>

  This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

  **Required?**
  false

  **Default Value**
  false

  **Accept Pipeline Input?**
  false

- **-Day** <RebootScheduleDays>

  For weekly schedules, the day of the week on which the scheduled reboot cycle starts (one of Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

  **Required?**
  false

  **Default Value**
  

  **Accept Pipeline Input?**
  false

- **-Enabled**<Boolean>

  

  **Required?**
  false

  **Default Value**
  

  **Accept Pipeline Input?**
  false
Boolean that indicates if the reboot schedule is to be enabled or disabled.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Frequency**<code>&lt;RebootScheduleFrequency&gt;</code>

Frequency with which this schedule runs (either Weekly or Daily).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-RebootDuration**<code>&lt;Int32&gt;</code>

Approximate maximum number of minutes over which the scheduled reboot cycle runs.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-StartTime**<code>&lt;TimeSpan&gt;</code>

Time of day at which the scheduled reboot cycle starts (HH:MM).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-WarningDuration**<code>&lt;Int32&gt;</code>

Time prior to the initiation of a machine reboot at which warning message is displayed in all user sessions on that machine. If the warning duration is zero then no message is displayed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-WarningMessage**<code>&lt;String&gt;</code>

Warning message displayed in user sessions on a machine scheduled for reboot. If the message is blank then no message is displayed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-WarningTitle**<code>&lt;String&gt;</code>

The window title used when showing the warning message in user sessions on a machine scheduled for reboot.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
-LoggingId<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

Input Type

Citrix.BrokerAdmin.SDK.RebootSchedule Reboot schedules may be specified through pipeline input.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerRebootSchedule -DesktopGroupName Accounting -WarningMessage "Save your work" -WarningDuration 10 -WarningTitle "WARNING: Reboot pending"
Sets the reboot schedule for the desktop group named Accounting to display a message with the title "WARNING: Reboot pending" and body "Save your work" ten minutes prior to rebooting each machine. The message is displayed in every user session on that machine.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerRebootSchedule -Frequency Weekly | Set-BrokerRebootSchedule -Day Friday
Sets all weekly reboot schedules to run on Friday.
Set-BrokerRemotePCAccount

Oct 10, 2013
Modify one or more RemotePCAccounts.

Syntax


Detailed Description

Modify one or more RemotePCAccounts.

Related topics

Get-BrokerRemotePCAccount
New-BrokerRemotePCAccount
Remove-BrokerRemotePCAccount

Parameters

- **InputObject** <RemotePCAccount[]>

Specifies the RemotePCAccounts to modify.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>False</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AllowSubfolderMatches**<Boolean>

When true a machine matches this RemotePCAccount if the AD computer is in the container specified by the OU property, or within a child container of the OU.
When false the AD computer object only matches if it is directly in the AD container specified by the OU property.

This property is not meaningful when OU has the special value ‘any’.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-MachinesExcluded</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-MachinesIncluded</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-MachinesExcluded<String[]>-  
MachinesExcluded specifies a set of strings that can include asterisk wildcards. If a machine name matches any entries in MachinesExcluded then it cannot match with this RemotePCAccount regardless of whether there is a MachinesIncluded match.

Matches are performed against the domain name joined with the machine name by a backslash (DOMAIN\MACHINE), e.g.:

DOMAIN1\M*  
DOMAIN*\M*  
*\M*

-OU<String>-  
Specifies the DN of an AD container, or has the special value ‘any’.
When an AD container is specified a machine may only match with the RemotePCAccount when the AD computer object is located relative to the OU.

When 'any' is specified the location of the AD computer object is ignored for purposes of matching this RemotePCAccount. The machine must still meet the MachinesIncluded and MachinesExcluded filters for a match to occur.

In the event that a machine matches with multiple RemotePCAccounts then the RemotePCAccount OU with the longest canonical name takes precedence. The special 'any' OU is treated as lowest priority.

Note that the OU value of every RemotePCAccount must be unique, and this includes only one 'any' entry being permitted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId<Guid>**

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress<String>**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.BrokerAdmin.SDK.RemotePCAccount You can pipe the RemotePCAccounts to be modified into this cmdlet.

**Return Values**

None or Citrix.BrokerAdmin.SDK.RemotePCAccount

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.RemotePCAccount object.
Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerRemotePCAccount | Set-BrokerRemotePCAccount -MachinesExcluded @("DOMAIN42\*")
Make all RemotePCAccounts filter out machines from DOMAIN42.
Set-BrokerSession

Oct 10, 2013
Sets properties of a session.

Syntax


Detailed Description

The Set-BrokerSession cmdlet sets properties on a session or set of sessions. You can specify a single session by Uid or multiple session instances can be passed to the command by piping or using the -InputObject parameter.

Related topics

Get-BrokerSession

Parameters

- **InputObject**<Session[]>

  The session instances whose properties you want to set.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **PassThru**<SwitchParameter>

  This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Hidden**<Boolean>

  Changes whether the session is hidden or not. Hidden sessions are treated as though they do not exist when brokering sessions; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

  A session may be hidden automatically by the system when an attempt is made to reconnect to a session that is unreachable due to, for example, the failure of a VM’s hypervisor. This allows a new session to be brokered provided that other machines in the same desktop group are still available. If the original session still exists after the hypervisor is restored then it must be unhidden to allow the user to reconnect to it.
Only sessions for shared desktops or applications can be hidden or unhidden. Private desktop or application sessions can never be hidden.

<table>
<thead>
<tr>
<th>-LoggingId&lt;Guid&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Required?</strong></td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
</tr>
</tbody>
</table>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>-AdminAddress&lt;String&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Required?</strong></td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
</tr>
</tbody>
</table>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.Session object.

**Examples**

------------- EXAMPLE 1 ---------------

C:\PS> $sessions = Get-BrokerSession -User ACCOUNTS\JohnSmith -Hidden $true
C:\PS> $sessions | Set-BrokerSession -Hidden $false

Finds all sessions in the site for user John Smith that have been hidden, and makes them available for reconnection again.
Set-BrokerSessionMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for Session

Syntax

Set-BrokerSessionMetadata [-SessionId] <Int64> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] []<CommonParameters>

Set-BrokerSessionMetadata [-SessionId] <Int64> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] []<CommonParameters>

Set-BrokerSessionMetadata [-SessionId] <Int64> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] []<CommonParameters>


Detailed Description

The Set-BrokerSessionMetadata cmdlet creates/updates metadata key-value pairs for Session. The Session can be specified by InputObject or piping.

Related topics

Parameters

- **SessionId**<Int64>

  Specifies the Session object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **InputObject**<Session[]>

  Specifies the Session objects whose Metadata is to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>
-**Map**<PSObject>

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Name**<String>

Specifies the name of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**Value**<String>

Specifies the value of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-**PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
**Accept Pipeline Input?**

false

---

**-AdminAddress**

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

---

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerSession You can pipe the Session to hold the new or updated metadata.

---

**Return Values**

None or Citrix.BrokerAdmin.SDK.BrokerSession

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerSession object.

---

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerSessionMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"
This command creates/updates the Metadata "MyMetadataName" key-value pair for the Session whose instance is pointed by $obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerSession | Set-BrokerSessionMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the Session in the site
Set-BrokerSharedDesktop

Oct 10, 2013
Change the settings of a shared desktop.

Syntax


Detailed Description

Shared desktops are automatically created when a machine is added to a desktop group with a DesktopKind of ‘Shared’, and these inherit default properties. Use Set-BrokerSharedDesktop to change the configuration settings of an existing shared desktop.

To specify shared desktops, you can choose whether to update by machine name or by passing a SharedDesktop or an array of SharedDesktop objects. You can also use the Uid or an array of Uids instead.

Most properties of a shared desktop cannot be modified as these contain status information; for example DNSName, RegistrationState, and OSVersion. You can change only the maintenance mode setting with this cmdlet.

Many of the properties that can be set with Set-BrokerSharedDesktop can be set by using Set-BrokerMachine (e.g. InMaintenanceMode). Using the Set-BrokerMachine cmdlet, where possible, is the preferred behaviour.

See about_Broker_Desktops for more information about desktops.

Related topics

Get-BrokerSharedDesktop

Parameters

- **-InputObject** <SharedDesktop[]>

  Specifies the desktop or array of desktops to modify. You can also use an integer Uid of the desktop instead.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-MachineName** <String>

  Specifies the desktop to modify using its machine name (in the form 'domain\machine').

<table>
<thead>
<tr>
<th>Required?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
</tr>
<tr>
<td>Default Value</td>
</tr>
<tr>
<td>-------------------------------</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

**-PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-InMaintenanceMode**<Boolean>

Changes the maintenance mode setting of a desktop. When a desktop is in maintenance mode, it is not included as a candidate when brokering new sessions, and it does not participate in automatic power management (idle pool); however, it still responds to explicit power operations.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Input Type
Citrix.BrokerAdmin.SDK.SharedDesktop You can pipe SharedDesktop objects into this cmdlet instead of on the command line with the -InputObject parameter.

Return Values
None or Citrix.BrokerAdmin.SDK.SharedDesktop
This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.SharedDesktop object.

Examples

-------------- EXAMPLE 1 --------------
C:\PS> Set-BrokerSharedDesktop DOMAIN\Machine1 -InMaintenanceMode $true
Put the Machine1 shared desktop into maintenance mode.

-------------- EXAMPLE 2 --------------
C:\PS> Get-BrokerSharedDesktop -InMaintenanceMode $true | Set-BrokerSharedDesktop -InMaintenanceMode $false
Bring all shared desktops currently in maintenance mode back into normal service.
Set-BrokerSite

Oct 10, 2013
Changes the overall settings of the current XenDesktop broker site.

Syntax


Detailed Description

The Set-BrokerSite cmdlet modifies properties of the current broker site.

The broker site is a top-level, logical representation of the XenDesktop site, from the perspective of the brokering services running within the site. It defines various site-wide default attributes used by the brokering services.

A XenDesktop installation has only a single broker site instance.

Related topics

Get-BrokerSite
New-BrokerIcon

Parameters

-PassThru<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-BaseOU<Guid>

Changes the objectGUID property identifying the base OU in Active Directory used for desktop registrations. For sites using only registry-based discovery (the default) this value is $null.

Any desktop attempting to register through a different OU from the one specified here is rejected. Note that desktops configured for registry-based discovery can register with the site, even if a BaseOU value is specified.

Information held in Active Directory is not modified by changing this value.

Typically, this property is changed only by using the Set-ADControllerDiscovery.ps1 script.
Default Value: false

-ColorDepth<br />
Changes the default color depth for new desktop groups, if no color depth is specified explicitly when a group is created. Changing this default has no impact on the color depths used already by existing groups.

Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

Required: false

DesktopGroupIconUid<br />
Changes the default desktop icon used for new desktop groups if no icon is specified explicitly when a group is created. Changing this default has no impact on the icons used already by existing groups.

The specified icon must already have been added to the site using New-BrokerIcon.

Required: false

DnsResolutionEnabled<br />
Changes whether ICA files returned by a broker service to a user device contain the numeric IP address or the DNS name of the desktop machine to which a session should be established.

With the default value ($false), ICA files will always contain a numeric IP address. To have DNS names appear in the ICA files, set the value to $true.

Even when DNS resolution is enabled ($true), IP addresses may still appear in ICA files. The reasons for this include, for example, that the broker service is unable to obtain a DNS name for the target machine, or that Web Interface is configured to always use numeric IP addresses in this context.

Required: false

SecureIcaRequired<br />

Changes the default SecureICA usage requirements for new desktop groups if no SecureICA setting is specified explicitly when a group is created. Changing this default has no impact on the SecureICA usage requirements of existing groups.

<table>
<thead>
<tr>
<th>-TrustRequestsSentToTheXmlServicePort</th>
<th>Boolean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Changes whether the XML Service (as used by Web Interface) implicitly trusts the originator of requests it receives, or whether it fully authenticates them.</td>
<td></td>
</tr>
<tr>
<td>With the default value ($false), full authentication checks are performed. However, you must enable this setting ($true) to allow support for &quot;Pass-through&quot; authentication, and/or connections routed through Access Gateway.</td>
<td></td>
</tr>
<tr>
<td>If this setting is enabled, you must ensure that controllers running the brokering services are securely firewalled.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-LoggingId</th>
<th>Guid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-AdminAddress</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input Type</th>
<th></th>
</tr>
</thead>
</table>
None You cannot pipe input into this cmdlet.

Return Values

None or Citrix.BrokerAdmin.SDK.Site

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.Site object.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerSite -ColorDepth SixteenBit
Specifies that any new desktop groups created, where a color depth value is not specified, default to using 16-bit color depth for user sessions to desktops or applications within that group.
Set-BrokerSiteMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for Site

Syntax

Set-BrokerSiteMetadata -Map <PSObject> [[-InputObject] <Site[]>] [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerSiteMetadata -Name <String> -Value <String> [[-InputObject] <Site[]>] [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerSiteMetadata -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerSiteMetadata -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerSiteMetadata -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerSiteMetadata cmdlet creates/updates metadata key-value pairs for Site. The Site can be specified by InputObject or piping.

Related topics

Parameters

- **Map**<PSObject>

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name**<String>

Specifies the name of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
**-Value**<String>

Specifies the value of the Metadata member to be created/updated

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-InputObject**<Site[]>

Specifies the Site objects whose Metadata is to be created/updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-PassThru**<SwitchParameter>

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1495
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
--- | ---
Accept Pipeline Input? | false

Input Type
Citrix.BrokerAdmin.SDK.BrokerSite You can pipe the Site to hold the new or updated metadata.

Return Values
None or Citrix.BrokerAdmin.SDK.BrokerSite

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerSite object.

Examples
-------------------------- EXAMPLE 1 --------------------------
C:\PS> Set-BrokerSiteMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates the Metadata "MyMetadataName" key-value pair for the Site

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-BrokerSite | Set-BrokerSiteMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234"
Set-BrokerTagMetadata

Oct 10, 2013
Creates/Updates metadata key-value pairs for Tag

Syntax

Set-BrokerTagMetadata [-TagId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerTagMetadata [-TagId] <Int32> -Name <String> -Value <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerTagMetadata [-TagId] <Int32> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-BrokerTagMetadata [-InputObject] <Tag[]> -Map <PSObject> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-BrokerTagMetadata cmdlet creates/updates metadata key-value pairs for Tag. The Tag can be specified by InputObject or piping.

Related topics

Parameters

- **-TagId**<Int32>

  Specifies the Tag object whose Metadata is to be created/updated by ID.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-InputObject**<Tag[]>

  Specifies the Tag objects whose Metadata is to be created/updated.

https://docs.citrix.com   © 1999-2017 Citrix Systems, Inc. All rights reserved.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TagName</strong>: <strong>String</strong></td>
<td>Specifies the Tag object whose Metadata is to be created/updated by name.</td>
</tr>
<tr>
<td><strong>Name</strong>: <strong>String</strong></td>
<td>Specifies the name of the Metadata member to be created/updated</td>
</tr>
<tr>
<td><strong>Value</strong>: <strong>String</strong></td>
<td>Specifies the value of the Metadata member to be created/updated</td>
</tr>
<tr>
<td><strong>Map</strong>: <strong>PSObject</strong></td>
<td>Specifies a hashtable containing name/value pairs to be used to create or update Metadata members</td>
</tr>
</tbody>
</table>

- **PassThru**: **SwitchParameter**

This cmdlet does not generate any output, unless you use the `PassThru` parameter, in which case it returns the affected record.
### -LoggingId <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

### -AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

**Input Type**

Citrix.BrokerAdmin.SDK.BrokerTag You can pipe the Tag to hold the new or updated metadata.

**Return Values**

None or Citrix.BrokerAdmin.SDK.BrokerTag

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.BrokerAdmin.SDK.BrokerTag object.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-BrokerTagMetadata -InputObject $obj-Uid -Name "MyMetadataName" -Value "1234"

This command creates/updates the Metadata "MyMetadataName" key-value pair for the Tag whose instance is pointed by Obj-Uid

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-BrokerTag | Set-BrokerTagMetadata -Name "MyMetadataName" -Value "1234"
This command creates/updates metadata key "MyMetadataName" with the value "1234" for all the Tag in the site

---------------------------- EXAMPLE 3 -----------------------------

C:\PS> @{ 'name1' = 'value1'; 'name2' = 'value2' } | Set-BrokerTagMetadata 'objname'
This command creates/updates two metadata keys "name1" and "name2" with the values "value1" and "value2" respectively for the Tag in the site whose name is 'objname'
Start-BrokerCatalogPvdImagePrepare

Oct 10, 2013
Start the PVD Image prepare process in the Broker for the machines in the specified catalog(s).

Syntax


Detailed Description

The Start-BrokerCatalogPvdImagePrepare cmdlet instructs the Broker to request the Personal VDisk (PVD) preparation process for all of the machines in the specified catalog(s). The process begins when each machine is subsequently powered off, at which point brokering of user desktop sessions is suspended as well as regular machine power operations until the process completes. Only catalogs with a PersistUserChanges value of OnPvd are supported by this cmdlet.

Related topics

Start-BrokerMachinePvdImagePrepare

Parameters

- **-InputObject** <Catalog[]>

  The catalog(s) holding the machines on which to start the PVD Image Preparation process.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
<td>true</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress** <String>

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.
Required? | false
--- | ---
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
Accept Pipeline Input? | false

### Input Type

**Citrix.BrokerAdmin.SDK.Catalog** You can pipe in the catalogs on which to start the PVD image preparation process.

### Return Values

None

### Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerCatalog 'MyCatalog' | Start-BrokerCatalogPvdImagePrepare
Instruct the Broker that the Personal VDisk preparation process will start the next time the machines in the specified catalog are powered off.
Start-BrokerMachinePvdImagePrepare

Oct 10, 2013
Start the PVD Image prepare process in the Broker for the specified machine(s).

Syntax


Detailed Description

The Start-BrokerMachinePvdImagePrepare cmdlet instructs the Broker to request the Personal VDisk (PVD) preparation process for the specified machine(s). The process begins when each machine is subsequently powered off, at which point brokering of user desktop sessions is suspended as well as regular machine power operations until the process completes. Only machines in catalogs with a PersistUserChanges value of OnPvd are supported by this cmdlet.

Related topics

Start-BrokerCatalogPvdImagePrepare

Parameters

- **InputObject**<Machine[]>

The machine(s) to start the PVD Image Preparation process on.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.
### Required?
false

### Default Value
Localhost. Once a value is provided by any cmdlet, this value will become the default.

### Accept Pipeline Input?
false

---

**Input Type**

*Citrix.BrokerAdmin.SDK.Machine* You can pipe in the machines to start the PVD image preparation process on.

---

**Return Values**

None

---

**Examples**

---------- **EXAMPLE 1**  ----------

```
```

Instruct the Broker that the Personal VDisk preparation process will start the next time the specified machine(s) are powered off.
Start-BrokerNaturalRebootCycle

Oct 10, 2013
Reboots all machines from the specified catalog when they are not in use.

Syntax

Start-BrokerNaturalRebootCycle [-InputObject] <Catalog[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Creating a natural reboot cycle for catalog ensures that all machines in the catalog are running the most recent image for the catalog and that all PvD image updates have been performed.

The machines are rebooted in a non disruptive manner, allowing machines that are in use to continue working and be restarted only after they become idle.

Related topics

None

Parameters

- **-InputObject** <Catalog[]>

Reboots all machines from this input catalog. The catalogs can be specified using UID values, name values (including wildcards) or catalog objects.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress** <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.
Input Type

Citrix.BrokerAdmin.SDK.Catalog Catalogs may be specified through pipeline input. The catalogs can be specified using UID values, name values (including wildcards) or catalog objects

Return Values

None

Notes

Natural reboot cycles do not apply to non-power managed and multi session catalogs

Examples

-------------------------- EXAMPLE 1 --------------------------

$c = Get-BrokerCatalog -Uid 1
    Start-BrokerNaturalRebootCycle -InputObject $c

The above code applies a natural reboot cycle on catalog with Id 1
Start-BrokerRebootCycle

Oct 10, 2013
Create and starts a reboot cycle for each desktop group that contains machines from the specified catalog.

Syntax


Detailed Description

The Start-BrokerRebootCycle cmdlet is used to create and start a reboot cycle for each desktop group that contains machines from the specified catalog. For a given desktop group, only the machines from the target catalog are rebooted and any machines from other catalogs are not rebooted.

Creating a reboot cycle for catalog ensures that all machines in the catalog are running the most recent image for the catalog and that all PVD image updates have been performed.

Related topics
Stop-BrokerRebootCycle
Get-BrokerRebootCycle

Parameters

-InputObject <Catalog[]>
Creates a reboot cycle for each desktop group that contains machines from this input catalog SDK object. The catalogs can be specified using UID values, name values (including wildcards) or catalog objects.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-RebootDuration <Int32>
Approximate maximum duration in minutes over which the reboot cycle runs.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-WarningDuration <Int32>
Time in minutes prior to a machine reboot at which a warning message is displayed in all user sessions on that machine. If the warning duration value is zero then no message is displayed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-WarningTitle <String>
The window title used when showing the warning message in user sessions on a machine scheduled for reboot.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
- **WarningMessage**: `String`

Warning message displayed in user sessions on a machine scheduled for reboot. If the message is blank then no message is displayed.

- **LoggingId**: `Guid`

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

- **AdminAddress**: `String`

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

Input Type

Citrix.BrokerAdmin.SDK.Catalog Catalogs may be specified through pipeline input. The catalogs can be specified using UID values, name values (including wildcards) or catalog objects

Return Values

none

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerCatalog -Name "SampleCatalog" | Start-BrokerRebootCycle -RebootDuration 240 -WarningMessage "Save your work" -WarningDuration 15

Starts a new reboot cycle for each desktop group containing machines from the catalog “SampleCatalog”. Each reboot cycle will have a duration of six hours. Fifteen minutes prior to rebooting a machine, the message “Save your work” will be displayed in each active user session.
Stop-BrokerRebootCycle

Oct 10, 2013
Cancels the specified reboot cycle.

Syntax

Stop-BrokerRebootCycle [-InputObject] <RebootCycle[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Stop-BrokerRebootCycle cmdlet is used to cancel the specified reboot cycle.

Related topics

Get-BrokerRebootCycle
Start-BrokerRebootCycle

Parameters

-**InputObject**<RebootCycle[]>

Cancels this reboot cycle.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
### Get-BrokerRebootCycle

**Default Value**

Localhost. Once a value is provided by any cmdlet, this value will become the default.

**Accept Pipeline Input?**

false

**Input Type**

Citrix.BrokerAdmin.SDK.RebootCycle

Reboot cycles may be specified through pipeline input.

**Return Values**

one

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-BrokerRebootCycle -CatalogUid 7 | Stop-BrokerRebootCycle

Cancels every reboot cycle for the catalog that has the Uid of 7.
Stop-BrokerSession

Oct 10, 2013
Stop or log off a session.

Syntax

Stop-BrokerSession [-InputObject] <Session[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters>

Detailed Description
Stops or logs off sessions.

Related topics
Disconnect-BrokerSession
Get-BrokerSession

Parameters

-**InputObject**<Session[]>-

Identifies the session(s) to terminate. This can be expressed as either a session Uid or a session object.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | false

Input Type
Citrix.BrokerAdmin.SDK.Session The sessions to stop can be piped into this cmdlet.

Return Values
None

Notes
This operation is non-blocking and returns before it completes. The operation, however, is unlikely to fail unless there are communication problems between the controller and the machine, if bad arguments are passed to the cmdlet itself or if the machine cannot successfully execute the operation.

The transient nature of sessions means that the list of session objects or UIDs supplied to Stop-BrokerSession could consist of valid and invalid sessions. Invalid sessions are detected and disregarded and the stop session operation is invoked on the valid sessions.

The system can fail to invoke the operation if the machine is not in an appropriate state or if there are problems in communicating with the machine. When an operation is invoked the system detects if the operation was initiated successfully or not by the machine. As this operation is non-blocking the system doesn't detect or report whether the operation ultimately succeeded or failed after its successful initialization on the machine.

Operation failures are reported through the broker SDK error handling mechanism (see about_Broker_ErrorHandling). In the event of errors the SdkErrorRecord error status is set to SessionOperationFailed and its error data dictionary is populated with the following entries:

- OperationsAttemptedCount - The number of operations attempted.
- OperationsFailedCount - The number of failed operations.
- OperationsSucceededCount - The number of successfully executed operations.
- UnresolvedSessionFailuresCount - The number of operations that failed due to invalid sessions being supplied.
- OperationInvocationFailuresCount - The number of operations that failed because they could not be invoked on the desktop.
- DesktopExecutionFailuresCount - The number of operations that failed because they could not be successfully executed by the desktop.

The SdkErrorRecord message will also display the number of attempted, failed and successful operations in the following format:


Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-BrokerSession -UserName MyDomain\MyAccount | Stop-BrokerSession
Stops all sessions for the user MyDomain\MyAccount.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> $desktop = Get-BrokerDesktop -DNSName MyMachine.MyDomain.com
C:\PS> Stop-BrokerSession $desktop.SessionUid
Stops the session on MyMachine.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Get-BrokerSession -Filter { SessionState -eq Disconnected -and SessionStateChangeTime -lt '-1'} | Stop-BrokerSession
Stop sessions that have been disconnected for more than one day.

-------------------------- EXAMPLE 4 --------------------------

C:\PS> trap [Citrix.Broker/Admin.SDK.SdkOperationException]
C:\PS> {
C:\PS>   write ($"Exception name = " + $_.Exception.GetType().FullName)
C:\PS>   write ($"SdkOperationException.Status = " + $_.Exception.Status)
C:\PS>   write ($"SdkOperationException.ErrorData=")
C:\PS>   $_.Exception.ErrorData
C:\PS>
C:\PS>   write ($"SdkOperationException.InnerException = " + $_.Exception.InnerException)
C:\PS>   $_.Exception.InnerException
C:\PS>   continue
C:\PS> }
C:\PS>
C:\PS> Stop-BrokerSession -InputObject 10,11,12
Trap and display error information.
Test-BrokerAccessPolicyRuleNameAvailable

Oct 10, 2013
Determine whether the proposed AccessPolicyRule Name is available for use.

Syntax

Test-BrokerAccessPolicyRuleNameAvailable [-Name <String[]>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

This cmdlet checks whether proposed AccessPolicyRule Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Related topics

Get-BrokerAccessPolicyRule
New-BrokerAccessPolicyRule
Rename-BrokerAccessPolicyRule

Parameters

- **-Name <String[]>**
  
  The AccessPolicyRule Name to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-AdminAddress <String>**

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

System.String You can pipe a string that contains the Name to test.

Return Values

https://docs.citrix.com
Citrix.BrokerAdmin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True" indicates the Name is available for use, and "False" if it is not available.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Test-BrokerAccessPolicyRuleNameAvailable -Name Test1
Checks whether the Name "Test1" is available.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Test-BrokerAccessPolicyRuleNameAvailable @("Test1","Test2","Test3")
Checks whether each of the specified names is available.
Test-BrokerAppAssignmentPolicyRuleNameAvailable

Oct 10, 2013
Determine whether the proposed AppAssignmentPolicyRule Name is available for use.

Syntax

Test-BrokerAppAssignmentPolicyRuleNameAvailable [-Name <String[]>] [-AdminAddress <String>] [CommonParameters>

Detailed Description

This cmdlet checks whether proposed AppAssignmentPolicyRule Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Related topics

Get-BrokerAppAssignmentPolicyRule
New-BrokerAppAssignmentPolicyRule
Rename-BrokerAppAssignmentPolicyRule

Parameters

-Name <String[]>

The AppAssignmentPolicyRule Name to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

System.String You can pipe a string that contains the Name to test.

Return Values
Citrix.BrokerAdmin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True" indicates the Name is available for use, and "False" if it is not available.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Test-BrokerAppAssignmentPolicyRuleNameAvailable -Name Test1
Checks whether the Name "Test1" is available.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Test-BrokerAppAssignmentPolicyRuleNameAvailable @("Test1","Test2","Test3")
Checks whether each of the specified names is available.
Test-BrokerAppEntitlementPolicyRuleNameAvailable

Oct 10, 2013
Determine whether the proposed AppEntitlementPolicyRule Name is available for use.

Syntax

Test-BrokerAppEntitlementPolicyRuleNameAvailable [-Name <String>] [-AdminAddress <String>] [CommonParameters]

Detailed Description
This cmdlet checks whether proposed AppEntitlementPolicyRule Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Related topics
Get-BrokerAppEntitlementPolicyRule
New-BrokerAppEntitlementPolicyRule
Rename-BrokerAppEntitlementPolicyRule

Parameters

- **Name<String>**
The AppEntitlementPolicyRule Name to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **AdminAddress<String>*
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

System.String You can pipe a string that contains the Name to test.

Return Values
Citrix.BrokerAdmin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True" indicates the Name is available for use, and "False" if it is not available.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Test-BrokerAppEntitlementPolicyRuleNameAvailable -Name Test1
Checks whether the Name "Test1" is available.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Test-BrokerAppEntitlementPolicyRuleNameAvailable @("Test1","Test2","Test3")
Checks whether each of the specified names is available.
Test-BrokerApplicationNameAvailable

Oct 10, 2013
Determine whether the proposed Application Name is available for use.

Syntax

Test-BrokerApplicationNameAvailable [-Name <String>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

This cmdlet checks whether proposed Application Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Related topics

Get-BrokerApplication
New-BrokerApplication
Rename-BrokerApplication

Parameters

-Name <String>
The Application Name to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

System.String You can pipe a string that contains the Name to test.

Return Values

Citrix.BrokerAdminSDK.NameAvailability
The cmdlet returns a result for each Name specified. An availability of "True" indicates the Name is available for use, and "False" if it is not available.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Test-BrokerApplicationNameAvailable -Name Test1
Checks whether the Name "Test1" is available.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Test-BrokerApplicationNameAvailable @("Test1", "Test2", "Test3")
Checks whether each of the specified names is available.
Test-BrokerAssignmentPolicyRuleNameAvailable

Oct 10, 2013
Determine whether the proposed AssignmentPolicyRule Name is available for use.

Syntax

Test-BrokerAssignmentPolicyRuleNameAvailable [-Name <String[]>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

This cmdlet checks whether proposed AssignmentPolicyRule Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Related topics

Get-BrokerAssignmentPolicyRule
New-BrokerAssignmentPolicyRule
Rename-BrokerAssignmentPolicyRule

Parameters

-Name <String[]>
The AssignmentPolicyRule Name to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

System.String You can pipe a string that contains the Name to test.

Return Values
Citrix.BrokerAdmin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True" indicates the Name is available for use, and "False" if it is not available.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Test-BrokerAssignmentPolicyRuleNameAvailable -Name Test1
Checks whether the Name "Test1" is available.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Test-BrokerAssignmentPolicyRuleNameAvailable @("Test1","Test2","Test3")
Checks whether each of the specified names is available.
Test-BrokerCatalogNameAvailable

Oct 10, 2013
Determine whether the proposed Catalog Name is available for use.

Syntax

Test-BrokerCatalogNameAvailable [-Name <String[]>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

This cmdlet checks whether proposed Catalog Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Related topics

Get-BrokerCatalog
New-BrokerCatalog
Rename-BrokerCatalog

Parameters

- **Name** <String[]>
  The Catalog Name to be tested.
  
  | Required? | true |
  | Default Value | |
  | Accept Pipeline Input? | true (ByValue, ByPropertyName) |

- **AdminAddress** <String>
  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.
  
  | Required? | false |
  | Default Value | localhost. Once a value is provided by any cmdlet, this value will become the default. |
  | Accept Pipeline Input? | false |

Input Type

System.String You can pipe a string that contains the Name to test.

Return Values

Citrix.BrokerAdmin.SDK.NameAvailability
The cmdlet returns a result for each Name specified. An availability of "True" indicates the Name is available for use, and "False" if it is not available.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Test-BrokerCatalogNameAvailable -Name Test1
Checks whether the Name "Test1" is available.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Test-BrokerCatalogNameAvailable @("Test1", "Test2", "Test3")
Checks whether each of the specified names is available.
Test-BrokerDBConnection

Oct 10, 2013
Tests whether a database is suitable for use by the Citrix Broker Service.

Syntax

Test-BrokerDBConnection [-DBConnection] <String> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix Broker Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to a Broker SDK cmdlet.

Related topics

Get-BrokerServiceStatus
Get-BrokerDBConnection
Set-BrokerDBConnection

Parameters

-DBConnection <String>

Specifies the database connection string to be tested by the currently selected Citrix Broker Service instance.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | false |

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1526
Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default.
---|---
Accept Pipeline Input? | false

Return Values

System.String

The Test-BrokerDBConnection cmdlet returns a string describing the status of the selected Citrix Broker Service instance that would result if the connection string were used with the Set-BrokerDBConnection cmdlet; the actual current status of the service is not changed. Possible values are:

-- OK:
The service instance is configured with a valid database and service schema. The service is operational.

-- DBUnconfigured:
No database connection string is set for the service instance.

-- DBRejectedConnection:
The database server rejected the logon from the service instance. This is typically caused by invalid credentials.

-- InvalidDBConfigured:
The specified database does not exist, is not visible to the service instance, or the service's schema within the database is invalid.

-- DBNotFound:
The specified database could not be located with the given connection string.

-- DBNewerVersionThanService:
The service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

-- DBOlderVersionThanService:
The service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

-- DBVersionChangeInProgress:
A database schema upgrade is in progress.

-- PendingFailure:
Connectivity between the service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

-- Failed:
Connectivity between the service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

-- Unknown:

Service status cannot be determined.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Test-BrokerDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;Trusted_Connection=True"
Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.
Test-BrokerDesktopGroupNameAvailable

Oct 10, 2013
Determine whether the proposed DesktopGroup Name is available for use.

Syntax

Test-BrokerDesktopGroupNameAvailable [-Name <String[]>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

This cmdlet checks whether proposed DesktopGroup Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Related topics

Get-BrokerDesktopGroup
New-BrokerDesktopGroup
Rename-BrokerDesktopGroup

Parameters

-Name <String[]>
The DesktopGroup Name to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

System.String You can pipe a string that contains the Name to test.

Return Values
The cmdlet returns a result for each Name specified. An availability of "True" indicates the Name is available for use, and "False" if it is not available.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Test-BrokerDesktopGroupNameAvailable -Name Test1
Checks whether the Name "Test1" is available.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Test-BrokerDesktopGroupNameAvailable @("Test1","Test2","Test3")
Checks whether each of the specified names is available.
Test-BrokerEntitlementPolicyRuleNameAvailable

Oct 10, 2013
Determine whether the proposed EntitlementPolicyRule Name is available for use.

Syntax
Test-BrokerEntitlementPolicyRuleNameAvailable [-Name <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
This cmdlet checks whether proposed EntitlementPolicyRule Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Related topics
Get-BrokerEntitlementPolicyRule
New-BrokerEntitlementPolicyRule
Rename-BrokerEntitlementPolicyRule

Parameters
-Name <String[]>
The EntitlementPolicyRule Name to be tested.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

Input Type
System.String You can pipe a string that contains the Name to test.

Return Values
Citrix.BrokerAdmin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True" indicates the Name is available for use, and "False" if it is not available.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Test-BrokerEntitlementPolicyRuleNameAvailable -Name Test1
Checks whether the Name "Test1" is available.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Test-BrokerEntitlementPolicyRuleNameAvailable @("Test1","Test2","Test3")
Checks whether each of the specified names is available.
Test-BrokerLicenseServer

Oct 10, 2013
Tests whether or not a license server can be used by the broker.

Syntax

Test-BrokerLicenseServer [-ComputerName] <String> [-AdminAddress <String>] [-Port <Int32>] [<CommonParameters>]

Detailed Description

Tests whether or not a given license server can be used by the broker.

Related topics

Get-BrokerSite
Set-BrokerSite

Parameters

-**ComputerName**<String>

The name of the license server to test (machine.domain).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>None</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Port**<Int32>

The port number to use on the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>27000</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Input Type
None You cannot pipe input into this cmdlet.

Return Values
System.String

Test-BrokerLicenseServer returns:

- 'Compatible' - the server is a compatible license server that can be used.
- 'Incompatible' - the server is an incompatible license server that can't be used.
- 'Inaccessible' - the server cannot be accessed. The server may be down, unreachable, or non-existent.
- 'InternalError' - the server can't be used due to an internal error. A required licensing component on the server may not be installed, configured, or working correctly.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Test-BrokerLicenseServer -LicenseServerAddress "machine.domain" 1234
Tests whether or not the license server "machine.domain" with port number 1234 can be used.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Test-BrokerLicenseServer -LicenseServerAddress "machine.domain"
Tests whether or not the license server "machine.domain" with port number 2700 can be used.
Test-BrokerMachineNameAvailable

Oct 10, 2013
Determine whether the proposed Machine MachineName is available for use.

Syntax

Test-BrokerMachineNameAvailable [-MachineName <String>][ -AdminAddress <String>] [<CommonParameters>]

Detailed Description

This cmdlet checks whether proposed Machine MachineName is available for use. It returns a record for each MachineName indicating the availability of that MachineName, with $true indicating that the MachineName is unused and available for use, or $false if it is not available.

Related topics

Get-BrokerMachine
New-BrokerMachine

Parameters

-MachineName <String[]>

The Machine MachineName to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

System.String You can pipe a string that contains the MachineName to test.

Return Values

Citrix.BrokerAdmin.SDK.NameAvailability
The cmdlet returns a result for each MachineName specified. An availability of "True" indicates the MachineName is available for use, and "False" if it is not available.

Examples

--------------- EXAMPLE 1 -----------------------------

C:\PS> Test-BrokerMachineNameAvailable -MachineName Test1
Checks whether the MachineName "Test1" is available.

--------------- EXAMPLE 2 -----------------------------

C:\PS> Test-BrokerMachineNameAvailable @("Test1","Test2","Test3")
Checks whether each of the specified names is available.
Test-BrokerPowerTimeSchemeNameAvailable

Oct 10, 2013
Determine whether the proposed PowerTimeScheme Name is available for use.

Syntax

```
Test-BrokerPowerTimeSchemeNameAvailable [-Name <String[]>] [-AdminAddress <String>] [CommonParameters>
```

Detailed Description

This cmdlet checks whether proposed PowerTimeScheme Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Related topics

- **Get-BrokerPowerTimeScheme**
- **New-BrokerPowerTimeScheme**
- **Rename-BrokerPowerTimeScheme**

Parameters

- **-Name <String[]>**
  
  The PowerTimeScheme Name to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

  | Accept Pipeline Input? | true (ByValue, ByPropertyName) |

- **-AdminAddress <String>**

  Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
</tbody>
</table>

  | Accept Pipeline Input? | false |

Input Type

```
System.String
```

You can pipe a string that contains the Name to test.

Return Values
Citrix.BrokerAdmin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True" indicates the Name is available for use, and "False" if it is not available.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Test-BrokerPowerTimeSchemeNameAvailable -Name Test1
Checks whether the Name "Test1" is available.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Test-BrokerPowerTimeSchemeNameAvailable @("Test1", "Test2", "Test3")
Checks whether each of the specified names is available.
Test-BrokerRemotePCAccountNameAvailable

Oct 10, 2013
Determine whether the proposed RemotePCAccount OU is available for use.

Syntax

Test-BrokerRemotePCAccountNameAvailable [-OU] <String>[] [-AdminAddress <String>] [CommonParameters]

Detailed Description

This cmdlet checks whether proposed RemotePCAccount OU is available for use. It returns a record for each OU indicating the availability of that OU, with $true indicating that the OU is unused and available for use, or $false if it is not available.

Related topics

Get-BrokerRemotePCAccount

New-BrokerRemotePCAccount

Parameters

-**OU**<String>[]>

The RemotePCAccount OU to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type
System.String You can pipe a string that contains the OU to test.

Return Values

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each OU specified. An availability of “True” indicates the OU is available for use, and “False” if
it is not available.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Test-BrokerRemotePCAccountNameAvailable -OU Test1
Checks whether the OU "Test1" is available.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Test-BrokerRemotePCAccountNameAvailable @("Test1","Test2","Test3")
Checks whether each of the specified names is available.
Update-BrokerImportedFTA

Oct 10, 2013
Imports or updates all of the file type associations for the specified worker.

Syntax

Update-BrokerImportedFTA -DesktopUids <Int32[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Imports or updates the file type associations from a specified worker machine.

File type association associates a file extension (such as ".txt") with an application (such as Notepad). In a Citrix environment file type associations on a user device can be configured so that when an user clicks on a document it launches the appropriate published application. This is known as "content redirection".

Imported file type associations are different from configured file type associations. Imported file type associations are lists of known file type associations for a given desktop group. Configured file type associations are those that are actually associated with published applications for the purposes of content redirection.

Initially the system is not aware of any extensions, and they must be imported by the Citrix administrator. To import or update file type associations from a worker machine, two criteria must be met:

- The worker machine must not be in use
- The worker machine must be in maintenance mode

For more information about putting a worker machine in maintenance mode, see the Set-BrokerPrivateDesktop and Set-BrokerSharedDesktop cmdlets.

Imported file type associations are grouped together based on the desktop group of the machine from which they were imported. All file types for a desktop group are deleted. There is no mechanism for deleting a subset imported file type associations for a specific desktop group.

If file type associations are imported more than once for a desktop group, for example, if this cmdlet is run twice for two workers belonging to the same desktop group, all existing imported file type associations for that desktop group are deleted and imported again.

Related topics

Get-BrokerImportedFTA
Remove-BrokerImportedFTA

Parameters

-DesktopUids<Int32[]>

Imports or updates the file type associations from the specified desktop. The desktop must belong to a desktop group of the Private or Shared desktop kind.
-**LoggingId**<Guid>

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Int32[] An array of Uids for desktops can be supplied as input.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $desktop = Get-BrokerSharedDesktop -MachineName "ACME\Worker1"
C:\PS> Set-BrokerSharedDesktop $desktop -InMaintenanceMode $true
C:\PS> Update-BrokerImportedFTA -DesktopUids $desktop.Uid
C:\PS> Set-BrokerSharedDesktop $desktop -InMaintenanceMode $false

Gets an object for the worker machine named "Worker1" in the "ACME" domain, and ensures no users can connect to it, before importing the file type associations from that desktop and re-enabling it.
Update-BrokerNameCache

Oct 10, 2013
Performs administrative operations on the user and machine name cache.

Syntax

Update-BrokerNameCache [-Machines] [-Users] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Triggers an immediate asynchronous refresh of the name cache. This may be useful to ensure up-to-date name information is present in the cache after user and/or machine accounts are known to have changed and you need to see those changes immediately instead of waiting for the periodic automatic refresh.

The Broker Service maintains a cache of the names of users and machines in use by the site. By default, name information is obtained periodically from Active Directory and the cache refreshed automatically.

During normal usage, you should not need to perform administrative operations on the name cache.

For users/groups, the following name information is cached:

Windows name (DOMAIN\user)

User Principal Name or 'UPN' (user@upndomain)

Full Name or 'Common Name' (typically a user's full name)

For machines, the following name information is cached:

Windows name (DOMAIN\machine)

DNS name (machine.dnsdomain)

Related topics

Parameters

-Machines <SwitchParameter>

Triggers an asynchronous refresh of all cached machine name information.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Users <SwitchParameter>

Triggers an asynchronous refresh of all cached user name information.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
</table>
### AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snapin will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

#### Input Type

None You cannot pipe input into this cmdlet.

#### Return Values

None

#### Notes

For some user accounts, for example, the built-in domain administrator, the UPN and/or Full Name values may not be available because they are not typically specified within Active Directory.

For group accounts, UPN and Full Name values are not available because they are not applicable or not specified within Active Directory.

The DNS name information for a machine is obtained from Active Directory and not from the DNS sub-system. If a machine has only recently been configured, the DNS information may not be available initially.

#### Examples

---------- EXAMPLE 1 ----------

C:\PS> Update-BrokerNameCache -Machines
Triggers an immediate asynchronous refresh of all machine name information held within the name cache.

---------- EXAMPLE 2 ----------

C:\PS> Update-BrokerNameCache -Machines -Users
Triggers an immediate asynchronous refresh of all machine and user name information held within the name cache.
# Overview

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConfigConfigurationSnapin</td>
<td>The Configuration service PowerShell snap-in provides administrative</td>
</tr>
<tr>
<td>Config Filtering</td>
<td>Describes the common filtering options for XenDesktop cmdlets.</td>
</tr>
</tbody>
</table>

## Cmdlets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-ConfigRegisteredServiceInstanceMetadata</td>
<td>Adds metadata on the given ServiceInstance.</td>
</tr>
<tr>
<td>Add-ConfigServiceGroupMetadata</td>
<td>Adds metadata on the given ServiceGroup.</td>
</tr>
<tr>
<td>Export-ConfigFeatureTable</td>
<td>Returns the current feature table.</td>
</tr>
<tr>
<td>Get-ConfigDBConnection</td>
<td>Gets the database string for the specified data store used by the</td>
</tr>
<tr>
<td></td>
<td>Configuration Service.</td>
</tr>
<tr>
<td>Get-ConfigDBSchema</td>
<td>Gets a script that creates the Configuration Service database schema for</td>
</tr>
<tr>
<td></td>
<td>the specified data store.</td>
</tr>
<tr>
<td>Get-ConfigDBVersionChangeScript</td>
<td>Gets a script that updates the Configuration Service database schema.</td>
</tr>
<tr>
<td>Get-ConfigEnabledFeature</td>
<td>Lists features of the site that are enabled.</td>
</tr>
<tr>
<td>Get-ConfigInstalledDBVersion</td>
<td>Gets a list of all available database schema versions for the Configuration</td>
</tr>
<tr>
<td></td>
<td>Service.</td>
</tr>
<tr>
<td>Get-ConfigLicensingModel</td>
<td>Lists the supported licensing models.</td>
</tr>
<tr>
<td>Get-ConfigLocalData</td>
<td>Gets the service local data.</td>
</tr>
<tr>
<td>Get-ConfigProduct</td>
<td>Lists the site's supported product names and codes.</td>
</tr>
<tr>
<td>Get-ConfigProductEdition</td>
<td>Lists the supported product editions.</td>
</tr>
<tr>
<td>Get-ConfigProductFeature</td>
<td>Lists the supported features.</td>
</tr>
<tr>
<td>Get-ConfigProductVersion</td>
<td>Lists the supported product versions.</td>
</tr>
<tr>
<td>Get-ConfigRegisteredServiceInstance</td>
<td>Gets the service instances that are registered in the directory.</td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Get-ConfigService</td>
<td>Gets the service record entries for the Configuration Service.</td>
</tr>
<tr>
<td>Get-ConfigServiceAddedCapability</td>
<td>Gets any added capabilities for the Configuration Service on the controller.</td>
</tr>
<tr>
<td>Get-ConfigServiceGroup</td>
<td>Gets the service groups that match the parameters supplied.</td>
</tr>
<tr>
<td>Get-ConfigServiceInstance</td>
<td>Gets the service instance entries for the Configuration Service.</td>
</tr>
<tr>
<td>Get-ConfigServiceStatus</td>
<td>Gets the current status of the Configuration Service on the controller.</td>
</tr>
<tr>
<td>Get-ConfigSite</td>
<td>Gets the site.</td>
</tr>
<tr>
<td>Import-ConfigFeatureTable</td>
<td>Sets the feature table of the site.</td>
</tr>
<tr>
<td>Register-ConfigServiceInstance</td>
<td>Allows the registration of a service instance.</td>
</tr>
<tr>
<td>Remove-ConfigRegisteredServiceInstanceMetadata</td>
<td>Removes metadata from the given ServiceInstance.</td>
</tr>
<tr>
<td>Remove-ConfigServiceGroup</td>
<td>Removes service groups.</td>
</tr>
<tr>
<td>Remove-ConfigServiceGroupMetadata</td>
<td>Removes metadata from the given ServiceGroup.</td>
</tr>
<tr>
<td>Remove-ConfigServiceMetadata</td>
<td>Removes metadata from the given Service.</td>
</tr>
<tr>
<td>Remove-ConfigSiteMetadata</td>
<td>Removes metadata from the given Site.</td>
</tr>
<tr>
<td>Reset-ConfigServiceGroupMembership</td>
<td>Reloads the access permissions and configuration service locations for the Configuration Service.</td>
</tr>
<tr>
<td>Set-ConfigDBConnection</td>
<td>Configures a database connection for the Configuration Service.</td>
</tr>
<tr>
<td>Set-ConfigRegisteredServiceInstance</td>
<td>Updates a service instance.</td>
</tr>
<tr>
<td>Set-ConfigRegisteredServiceInstanceMetadata</td>
<td>Adds or updates metadata on the given ServiceInstance.</td>
</tr>
<tr>
<td>Set-ConfigServiceGroupMetadata</td>
<td>Adds or updates metadata on the given ServiceGroup.</td>
</tr>
<tr>
<td>Set-ConfigServiceMetadata</td>
<td>Adds or updates metadata on the given Service.</td>
</tr>
<tr>
<td>Set-ConfigSite</td>
<td>Changes the overall settings of the site.</td>
</tr>
<tr>
<td>Set-ConfigSiteMetadata</td>
<td>Adds or updates metadata on the Site.</td>
</tr>
<tr>
<td>Test-ConfigDBConnection</td>
<td>Tests a database connection for the Configuration Service.</td>
</tr>
<tr>
<td>Test-ConfigServiceInstanceAvailability</td>
<td>Tests whether the supplied service instances are responding to requests.</td>
</tr>
<tr>
<td>Unregister-</td>
<td>Removes a service instance from the Configuration Service registry.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>

The Configuration service PowerShell snap-in provides administrative functions for the Configuration service.

All commands in this snap-in have 'Config' in their name.

The Configuration service PowerShell snap-in enables both local and remote administration of the Configuration service. It provides facilities to store details about other services that are used in the XenDesktop deployment. All the services in the XenDesktop deployment use the Configuration service as a directory to locate other services with which they need to communicate. The directory publishes a list of all the services, the communication types they accept, and the facilities they offer.

The snap-in provides the following main entities:

Site Metadata

Metadata for the entire XenDesktop deployment. This metadata is not used directly by any of the XenDesktop services, but can be used by third parties to store information for other purposes.

Registered Service Instances

Service instance items that have been retrieved from the available services in the XenDesktop deployment and held in a directory in the Configuration service. Each physical service can host a collection of service instances to provide different facilities.

Service Groups

A collection of service instances that are considered equivalent. Service instances that are registered in the same service group provide the same state and offer the same functionality.

Only one service group of each type is supported. For example, there
must not be more than one service group with a ServiceType of 'Config'.

Site and Features

The configuration site is a top-level, logical representation of the XenDesktop site, from the perspective of the configuration services running within the site.
about_Config_Filtering

Oct 10, 2013

TOPIC

XenDesktop - Advanced Dataset Filtering

SHORT DESCRIPTION

Describes the common filtering options for XenDesktop cmdlets.

LONG DESCRIPTION

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get-cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

    Get-Noun > Where { $_.Size = 'Small' } | Sort 'Date' | Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get-cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

    Get-Noun -Size 'Small' -SortBy 'Date' -MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

- **-MaxRecordCount <int>**

  Specifies the maximum number of results to return.
  For example, to return only the first nine results use:

  ```powershell
  Get-Noun -MaxRecordCount 9
  ```

  If not specified, only the first 250 records are returned, and if more are available, a warning is produced:
WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

To retrieve all records, specify a large number for -MaxRecordCount.
As the value is an integer, you can use the following:

```powershell
Get-<Noun> -MaxRecordCount [int]:MaxValue
```

- **ReturnTotalRecordCount [<SwitchParameter>]**

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```powershell
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
```

Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<< -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo : OperationStopped: (:) [Get-<Noun>], PartialDataException
+ FullyQualifiedErrorId: PartialData,Citrix.<SDKName>.SDK.Get<Noun>

The count can be accessed using the TotalAvailableResultCount property:

```
$count = $error[0].TotalAvailableResultCount
```

- **Skip <int>**

Skips the specified number of records before returning results.
Also reduces the count returned by -ReturnTotalRecordCount.

- **SortBy <string>**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.
Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

- sortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values. Elements not mentioned are placed at the end in their numeric order.

For example, to sort by two different enums and then by the object id:

- sortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter 'Name -like "High*" -or (Priority -eq 1 -and Severity -ge 2)'

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

EXAMPLES

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company "citrix" -Product '[X]EN''
Get-<Noun> -Product "Xen" -Company "CITRIX"
Get-<Noun> -Filter { Company -eq 'Citrix' -and Product -like 'Xen' }
See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
Get-<Noun> -Company "Abc["]" # Matches Abc*
Get-<Noun> -Company "Abc"** # Matches Abc*
Get-<Noun> -Filter { Company -eq "Abc"** } # Matches Abc*
Get-<Noun> -Filter { Company -eq "A""B""C" } # Matches A"B"C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

```
Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```
Get-<Noun> -Filter 'Enabled'      # Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled' # Equivalent to 'Enabled -eq $false'
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```
Get-<Noun> -Shape [Shapes]:Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle" }
Get-<Noun> -Filter { Shape -like 'C*' }

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z" }
$d = [DateTime]"2010-08-23T12:30:00.0Z"
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge '-2' } # Two days ago
Get-<Noun> -Filter { StartTime -ge '-1:30' } # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge '-0:0:30' } # 30 seconds ago

ARRAY PROPERTIES
When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings. Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*" }
Get-<Noun> -Filter { Users -contains "Fred*" }

You can also use the singular form with -Filter to search using other operators:

# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter { User -eq "Frederick" }
# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter { User -lt 'F' }

COMPLEX EXPRESSIONS

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1554
When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

```
Get-<Noun> -Filter { Shape -eq 'Circle' -or Shape -eq 'Square' }
$shapes = 'Circle', 'Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }
```

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

```
Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue' -and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue' -and Shape -eq 'Circle') }
```

**PAGING**

The simplest way to page through data is to use the -Skip and -MaxRecordCount parameters. So, to read the first three pages of data with 10 records per page, use:

```
Get-<Noun> -Skip 0  -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 10 -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 20 -MaxRecordCount 10 <other filtering criteria>
```

You must include the same filtering criteria on each call, and ensure that the data is sorted consistently.

The above approach is often acceptable, but as each call performs an independent query, data changes can result in records being skipped or appearing twice. One approach to improve this is to sort by a unique id field and then start the search for the next page at the unique id after the last unique id of the previous page. For example:

```
# Get the first page
Get-<Noun> -MaxRecordCount 10 -SortBy SerialNumber

Serializable  ...
------------- ---
A120004
A120007
... 7 other records ...
A120900

# Get the next page
Get-<Noun> -MaxRecordCount 10 -Filter { FirstName -gt 'A120900' }

Serializable  ...
```
FILTER SYNTAX DEFINITION

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= "{" <ComponentList> "}"

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

          <Component>

          <Component> ::= <NotOperator> <Factor> |

          <Factor>

          <Factor> ::= "(" <ComponentList> ")" |

          <PropertyName> <ComparisonOperator> <Value> |

          <PropertyName>

<AndOrOperator> ::= "-and" | "-or"

<NotOperator> ::= "-not" | "!

<ComparisonOperator>

  ::= "-eq" | "-ne" | "-le" | "-ge" | "-lt" | "-gt" |
  "-like" | "-notlike" | "-contains" | "-notcontains" |
  "-in" | "-notin"

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

          <scalar variable> | <array variable> |
  "$null" | "$true" | "$false"
Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30" means 1 hour and 30 minutes ago.
Add-ConfigRegisteredServiceInstanceMetadata

Oct 10, 2013
Adds metadata on the given ServiceInstance.

Syntax

Add-ConfigRegisteredServiceInstanceMetadata [-ServiceInstanceUid] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Add-ConfigRegisteredServiceInstanceMetadata [-ServiceInstanceUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Add-ConfigRegisteredServiceInstanceMetadata [-InputObject] <ServiceInstance[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Add-ConfigRegisteredServiceInstanceMetadata [-InputObject] <ServiceInstance[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Provides the ability for additional custom data to be stored against given ServiceInstance objects. This cmdlet will not overwrite existing metadata on an object - use the Set-ConfigRegisteredServiceInstanceMetadata cmdlet instead.

Related topics

Set-ConfigRegisteredServiceInstanceMetadata
Remove-ConfigRegisteredServiceInstanceMetadata

Parameters

- **-ServiceInstanceUid** <Guid>

Id of the ServiceInstance

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-InputObject** <ServiceInstance[]>

Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** <String>

Specifies the property name of the metadata to be added. The property must be unique for the ServiceInstance specified. The property cannot contain any of the following characters \/:*?<>|[]()"'

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
-Value<String>

Specifies the value for the property.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-Map<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-LoggingId<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values

Citrix.Configuration.Sdk.Metadata

Add-ConfigRegisteredServiceInstanceMetadata returns an array of objects containing the new definition of the metadata.

\n Property <string>
\n Specifies the name of the property.
\n Value <string>
\n Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DuplicateObject
One of the specified metadata already exists.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------

c:s\PS> Add-ConfigRegisteredServiceInstanceMetadata -ServiceInstanceUid 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the ServiceInstance with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1560
Add-ConfigServiceGroupMetadata

Oct 10, 2013
Adds metadata on the given ServiceGroup.

Syntax
Add-ConfigServiceGroupMetadata [-ServiceGroupUid] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]
Add-ConfigServiceGroupMetadata [-ServiceGroupUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]
Add-ConfigServiceGroupMetadata [-InputObject] <ServiceGroup[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]
Add-ConfigServiceGroupMetadata [-InputObject] <ServiceGroup[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description
Provides the ability for additional custom data to be stored against given ServiceGroup objects. This cmdlet will not overwrite existing metadata on an object - use the Set-ConfigServiceGroupMetadata cmdlet instead.

Related topics
Set-ConfigServiceGroupMetadata
Remove-ConfigServiceGroupMetadata

Parameters

(ServiceGroupUid)<Guid>
Id of the ServiceGroup

Required? true
Default Value
Accept Pipeline Input? true (ByValue, ByPropertyName)

(InputObject)<ServiceGroup[]>
Objects to which the metadata is to be added.

Required? true
Default Value
Accept Pipeline Input? true (ByValue)

(Name)<String>
Specifies the property name of the metadata to be added. The property must be unique for the ServiceGroup specified. The property cannot contain any of the following characters \\
\/:.*?<>|{[\]}

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1561
Add-ConfigServiceGroupMetadata returns an array of objects containing the new definition of the metadata.
Property <string>

Specifies the name of the property.

Value <string>

Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

--------

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DuplicateObject
One of the specified metadata already exists.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
c:\PS> Add-ConfigServiceGroupMetadata -ServiceGroupId 4CECC26E-48E1-423F-A1F0-2A06DD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the ServiceGroup with the identifier '4CECC26E-48E1-423F-A1F0-2A06DD0805C'.
Export-ConfigFeatureTable

Oct 10, 2013
Returns the current feature table.

Syntax
Export-ConfigFeatureTable [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Related topics
Set-ConfigSite
Import-ConfigFeatureTable

Parameters
-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

Return Values
System.String

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Export-ConfigFeatureTable
<?xml version="1.0" encoding="utf-8">
<Product xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="FeatureTable.xsd">
  <Product Code="XDT" Name="XenDesktop">
    <Editions>
      <Edition>PLT</Edition>
      <Edition>ENT</Edition>
      <Edition>APP</Edition>
      <Edition>STD</Edition>
    </Editions>
    <VersionToBurninDates>
      <VersionToBurninDate Version="7.0" BurninDate="2013.0522" />
    </VersionToBurninDates>
    <LicensingModels>
      <LicensingModel>Concurrent</LicensingModel>
      <LicensingModel>UserDevice</LicensingModel>
    </LicensingModels>
    <DefaultLicensingModel>UserDevice</DefaultLicensingModel>
    <Features>
      <Feature Name="CustomRole">
        <BurninDate ForProductEdition="PLT" MustBe="AtLeast">2013.0522</BurninDate>
        <BurninDate ForProductEdition="ENT" MustBe="AtLeast">2013.0522</BurninDate>
      </Feature>
    </Features>
  </Product>
</Products>
<LicensingModels>
<DefaultLicensingModel>Concurrent</DefaultLicensingModel>
</LicensingModels>

<Features>

<Feature Name="CustomRole">
    <BurninDate ForProductEdition="PLT" MustBe="AtLeast">2013.0522</BurninDate>
    <BurninDate ForProductEdition="ENT" MustBe="AtLeast">2013.0522</BurninDate>
</Feature>

<Feature Name="ConfigurationLogging">
    <BurninDate ForProductEdition="PLT" MustBe="AtLeast">2013.0522</BurninDate>
    <BurninDate ForProductEdition="ENT" MustBe="AtLeast">2013.0522</BurninDate>
</Feature>

<Feature Name="SingleUserMode">
</Feature>

<Feature Name="MultiSessionDesktops">
    <BurninDate ForProductEdition="PLT" MustBe="AtLeast">2013.0522</BurninDate>
    <BurninDate ForProductEdition="ENT" MustBe="AtLeast">2013.0522</BurninDate>
</Feature>

<Feature Name="SingleSessionDesktops">
</Feature>

<Feature Name="MultiSessionApplications">
    <BurninDate ForProductEdition="PLT" MustBe="AtLeast">2013.0522</BurninDate>
    <BurninDate ForProductEdition="ENT" MustBe="AtLeast">2013.0522</BurninDate>
</Feature>

<Feature Name="SingleSessionApplications">
    <BurninDate ForProductEdition="PLT" MustBe="AtLeast">2013.0522</BurninDate>
    <BurninDate ForProductEdition="ENT" MustBe="AtLeast">2013.0522</BurninDate>
</Feature>

<Feature Name="HistoricalMonitorData">
</Feature>

<Feature Name="CloudHostedMachines">
</Feature>

<Feature Name="HdxInsightIntegration">
</Feature>

<Feature Name="RemotePC">
</Feature>
</Features>

Returns the current feature table.
Get-ConfigDBConnection

Oct 10, 2013
Gets the database string for the specified data store used by the Configuration Service.

Syntax

Get-ConfigDBConnection [-AdminAddress <String>] [CommonParameters]

Detailed Description

Returns the database connection string for the specified data store.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

Related topics

Get-ConfigServiceStatus
Set-ConfigDBConnection
Test-ConfigDBConnection

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Return Values

system.string

The database connection string configured for the Configuration Service.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

NoDBConnections
The database connection string for the Configuration Service has not been specified.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ConfigDBConnection

Server=serverName\SQLExpress;Initial Catalog = databaseName; Integrated Security = True
Get the database connection string for the Configuration Service.
Get-ConfigDBSchema

Oct 10, 2013
Gets a script that creates the Configuration Service database schema for the specified data store.

Syntax

Get-ConfigDBSchema [-DatabaseName <String>] [-ServiceGroupName <String>] [-ScriptType <ScriptTypes>] [-LocalDatabase] [-Sid <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Gets SQL scripts that can be used to create a new Configuration Service database schema, add a new Configuration Service to an existing site, remove a Configuration Service from a site, or create a database server logon for a Configuration Service. If no Sid parameter is provided, the scripts obtained relate to the currently selected Configuration Service instance, otherwise the scripts relate to Configuration Service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied. The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to a Configuration SDK cmdlet. The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured. The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SMDCMD mode'. The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to Configuration Service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to Configuration Service roles

If ScriptType is Evict, the returned script contains:

- Removal of Configuration Service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

If the service uses two data stores they can exist in the same database. You do not need to configure a database before...
using this command.

Related topics

Set-ConfigDBConnection

Parameters

- **DatabaseName**<String>

  Specifies the name of the database for which the schema will be generated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

- **ServiceGroupName**<String>

  Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the Configuration services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

- **ScriptType**<ScriptTypes>

  Specifies the type of database script returned. Available script types are:

  Database

  Returns a full database script that can be used to create a database schema for the Configuration Service in a database instance that does not already contain a schema for this service. The DatabaseName and ServiceGroupName parameters must be specified to create a script of this type.

  Instance

  Returns a permissions script that can be used to add further Configuration services to an existing database instance that already contains the full Configuration service schema, associating the services to the Service Group. The Sid parameter can optionally be specified to create a script of this type.

  Login

  Returns a database logon script that can be used to add the required logon accounts to an existing database instance that contains the Configuration Service schema. This is used primarily when creating a mirrored database environment. The DatabaseName parameter must be specified to create a script of this type.
Evict

Retruns a script that can be used to remove the specified Configuration Service from the database entirely. The DatabaseName and Sid parameters must be specified to create a script of this type.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Database</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LocalDatabase**<SwitchParameter>

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for Configuration services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Sid**<String>

Specifies the SID of the controller on which the Configuration Service instance to remove from the database is running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.String

A string containing the required SQL script for application to a database.
Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

Create the database schema.
Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:
Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:
Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned.

Error Codes

GetSchemasFailed
The database schema could not be found.

ActiveDirectoryAccountResolutionFailed
The specified Active Directory account or Group could not be found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ConfigDBSchema -DatabaseName MyDB -ServiceGroupName MyServiceGroup > c:\ConfigSchema.sql
Get the full database schema for site data store of the Configuration Service and copy it to a file called 'c:\ConfigSchema.sql'.

This script can then be used to create the schema in a pre-existing database named 'MyDB' that does not already contain a Configuration Service site schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-ConfigDBSchema -DatabaseName MyDB -scriptType Login > c:\ConfigurationLogins.sql
Get the logon scripts for the Configuration Service.
Get-ConfigDBVersionChangeScript

Oct 10, 2013

Gets a script that updates the Configuration Service database schema.

Syntax

Get-ConfigDBVersionChangeScript -DatabaseName <String> -TargetVersion <Version> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns a database script that can be used to upgrade or downgrade the site or secondary schema for the Configuration Service from the current schema version to a different version.

Related topics

Get-ConfigInstalledDBVersion

Parameters

-DatabaseName <String>

Specifies the name of the database instance to which the update applies.

| Required? | true |
| Default Value | 
| Accept Pipeline Input? | false |

-TargetVersion <Version>

Specifies the version of the database you want to update to.

| Required? | true |
| Default Value | 
| Accept Pipeline Input? | false |

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |
Return Values

System.Management.Automation.PSObject

A PSObject containing the required SQL script for application to a database.

Notes

The PSObject returned by this cmdlet contains the following properties:

-- Script The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

-- NeedExclusiveAccess Indicates whether all services in the service group must be shut down during the update or not.

-- CanUndo Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any Configuration services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-ConfigServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned.

Error Codes

----------

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the Configuration Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------- EXAMPLE 1 --------------------

C:\PS> $update = Get-ConfigDBVersionChangeScript -DatabaseName MyDb -TargetVersion 1.0.75.0
C:\PS> $update.Script > update_75.sql

Gets an SQL update script to update the current schema to version 1.0.75.0. The resulting update_75.sql script is suitable for direct use with the SQL Server SQLCMD utility.
Get-ConfigEnabledFeature

Oct 10, 2013
Lists features of the site that are enabled.

Syntax

Get-ConfigEnabledFeature [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Related topics

Set-ConfigSite

Import-ConfigFeatureTable

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.String

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-ConfigEnabledFeature
Retrieves the list of enabled features for the site's current configuration.
Get-ConfigInstalledDBVersion

Oct 10, 2013

Gets a list of all available database schema versions for the Configuration Service.

Syntax

Get-ConfigInstalledDBVersion [-Upgrade] [-Downgrade] [-AdminAddress <String>] [CommonParameters>

Detailed Description

Returns the current version of the Configuration Service database schema, if no flags are set, otherwise returns versions for which upgrade or downgrade scripts are available and have been stored in the database.

Related topics

Parameters

- **Upgrade**<SwitchParameter>

  Specifies that only schema versions to which the current database version can be updated should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **Downgrade**<SwitchParameter>

  Specifies that only schema versions to which the current database version can be reverted should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
The Get-ConfigInstalledDbVersion command returns objects containing the new definition of the Configuration Service database schema version.

Major <Integer>
Minor <Integer>
Build <Integer>
Revision <Integer>

Notes
If the command fails, the following errors can be returned.

Error Codes
-----------
InvalidParameterCombination
Both the Upgrade and Downgrade flags were specified.
NoOp
The operation was successful but had no effect.
NoDBConnections
The database connection string for the Configuration Service has not been specified.
DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.
PermissionDenied
You do not have permission to execute this command.
AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.
CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ConfigInstalledDBVersion

Major Minor Build Revision
----- ----- ----- --------
5 6 0 0
Get the currently installed version of the Configuration Service database schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-ConfigInstalledDBVersion -Upgrade

Major Minor Build Revision
----- ----- ----- --------
6 0 0 0
Get the versions of the Configuration Service database schema for which upgrade scripts are supplied.
Get-ConfigLicensingModel

Oct 10, 2013
Lists the supported licensing models.

Syntax

Get-ConfigLicensingModel -ProductCode <String> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Related topics

Get-ConfigProduct
Get-ConfigSite
Set-ConfigSite
Import-ConfigFeatureTable

Parameters

-ProductCode <String>

The product code

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.String

The list of supported licensing models for the specified product code.

Notes
The Get-ConfigProduct cmdlet lists the available product codes.

The site object returned by the Get-ConfigSite cmdlet contains the currently configured product code.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-ConfigLicensingModel -ProductCode "XDS"
Retrieves the list of supported licensing models for product code "XDS".
Get-ConfigLocalData

Oct 10, 2013
Gets the service local data.

Syntax

Get-ConfigLocalData [-AdminAddress <String>] [CommonParameters]

Detailed Description

Get-ConfigLocalData returns service-and-controller-specific local data. This information is not site-wide, but rather controller-specific. The Configuration Service currently stores the controller product version as local data. The overall site version used by Studio is an aggregate of the product versions from each controller.

Related topics

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

This cmdlet does not accept pipeline input

Return Values

Citrix.Configuration.DataModel.LocalData

Contains the ControllerProductVersion field

Examples

------------------------- EXAMPLE 1 -------------------------

C:\PS> Get-ConfigLocalData

ControllerProductVersion
-----------
7.1
Gets the service local data.
Get-ConfigProduct

Oct 10, 2013
Lists the site's supported product names and codes.

Syntax

Get-ConfigProduct [-AdminAddress <String>] [CommonParameters>

Detailed Description

Related topics

Get-ConfigSite

Set-ConfigSite

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

PSObject

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-ConfigProduct
Lists the supported products by name and code.
Get-ConfigProductEdition

Oct 10, 2013
Lists the supported product editions.

Syntax
Get-ConfigProductEdition [-ProductCode] <String> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Related topics
Get-ConfigProduct
Get-ConfigSite
Set-ConfigSite
Import-ConfigFeatureTable

Parameters

-**ProductCode**<String>
The product code

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.String
The list of supported licensing models for the specified product code.

Notes
The Get-ConfigProduct cmdlet lists the available product codes.

The site object returned by Get-ConfigSite cmdlet contains the currently configured product code.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-ConfigProductEdition -ProductCode "XDS"
Retrieves the list of supported editions for XenDesktop.
Get-ConfigProductFeature

Oct 10, 2013
Lists the supported features.

Syntax

Get-ConfigProductFeature [-ProductCode] <String> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Lists the supported features. Use the Get-ConfigEnabledFeature command to determine which features are currently enabled.

Related topics

Get-ConfigProduct

Get-ConfigSite

Set-ConfigSite

Import-ConfigFeatureTable

Parameters

- **ProductCode**<String>

  The product code

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.String

The list of supported licensing models for the specified product code.
Notes

The Get-ConfigProduct cmdlet lists the available product codes.

The site object returned by Get-ConfigSite cmdlet contains the currently configured product code.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-ConfigProductFeature -ProductCode "XDS"
Retrieves the list of supported features for XenDesktop.
Get-ConfigProductVersion

Oct 10, 2013
Lists the supported product versions.

Syntax
Get-ConfigProductVersion [-ProductCode <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Related topics
Get-ConfigProduct
Get-ConfigSite
Set-ConfigSite
Import-ConfigFeatureTable

Parameters
-ProductCode <String>
The product code

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values
System.String
The list of supported licensing models for the specified product code.

Notes
The `Get-ConfigProduct` cmdlet lists the available product codes.

The site object returned by `Get-ConfigSite` cmdlet contains the currently configured product code.

Examples

-------------------------- EXAMPLE 1 --------------------------

```
C:\PS> Get-ConfigProductVersion -ProductCode "XDS"
```

 Retrieves the list of supported versions for XenDesktop.
Get-ConfigRegisteredServiceInstance

Oct 10, 2013
Gets the service instances that are registered in the directory.

Syntax


Detailed Description

Use this cmdlet to retrieve the service instances currently registered with the Configuration Service that match the parameters supplied. If no parameters are supplied, all the service instances are returned.

Related topics

Register-ConfigServiceInstance
Unregister-ConfigRegisteredServiceInstance
Set-ConfigRegisteredServiceInstance

Parameters

- **ServiceInstanceUid** <Guid>

  The unique identifier for the service instance.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ServiceGroupUid** <Guid>

  The unique identifier for the service group to which the service instance belongs.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ServiceGroupName** <String>

  The name for the service group to which the service instance belongs.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ServiceType</td>
<td>String</td>
<td>The service type for the service instance.</td>
</tr>
<tr>
<td>Address</td>
<td>String</td>
<td>The connection address for the service instance.</td>
</tr>
<tr>
<td>Binding</td>
<td>String</td>
<td>The binding for the service instance.</td>
</tr>
<tr>
<td>Version</td>
<td>Int32</td>
<td>The service instance version.</td>
</tr>
<tr>
<td>ServiceAccountSid</td>
<td>String</td>
<td>The AD account SID for the computer account that the computer hosting the service instance is running as.</td>
</tr>
</tbody>
</table>
### `-InterfaceType <String>`

The interface type for the service instance.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### `-Metadata <String>`

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: `-Metadata "abc:x*"` matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### `-Property <String[]>`

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### `-ReturnTotalRecordCount <SwitchParameter>`

See about_Configfiltering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
-**MaxRecordCount** `<Int32>`

See about `Configfiltering` for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Skip** `<Int32>`

See about `Configfiltering` for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**SortBy** `<String>`

See about `Configfiltering` for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Filter** `<String>`

See about `Configfiltering` for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress** `<String>`

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values

Citrix.Configuration.Sdk.ServiceInstance
This represents a service instance and has the following parameters:

ServiceGroupUid <Guid>
The unique identifier for the service group to which the service instance belongs.

ServiceGroupName <string>
The name of the service group to which the service instance belongs.

ServiceInstanceUid <Guid>
The unique identifier for the service instance.

ServiceType <string>
The type of the service group.

Address <string>
The contact address for the service instance.

Binding <string>
The binding to use for connections to the service instance.

Version <int>
The version of the service instance.

ServiceAccount <string>
The AD computer account for the computer that is providing the service instance.

ServiceAccountSid <string>
The AD computer account SID for the computer that is providing the service instance.

InterfaceType <string>
The interface type for the service instance.

Metadata <Citrix.Configuration.Sdk.Metadata[]>
The metadata for the service instance.

Notes
In the case of failure, the following errors can result.

Error Codes ----------- PartialData Only a subset of the available data was returned. CouldNotQueryDatabase The query
required to get the database was not defined. CommunicationError An error occurred while communicating with the service. DatabaseNotConfigured The operation could not be completed because the database for the service is not configured. InvalidFilter A filtering expression was supplied that could not be interpreted for this cmdlet. ExceptionThrown An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\>Get-ConfigRegisteredServiceInstance

Address            : http://MyServer.com:80/Citrix/MyContract/v1
Binding            : wcf_HTTP_kerb
InterfaceType      : SDK
Metadata           : {}
MetadataMap        : {}
ServiceAccount     : ENG\MyAccount$
ServiceAccountSid  : S-1-5-21-1155438255-2213498043-2452000591-1104
ServiceGroupName   : MyService
ServiceGroupUid    : 2b990d5a-bba9-413b-aa08-e104e67f89bc
ServiceInstanceUid : 8dc38b5a-3fbb-457c-b326-6c41c94c18d5
ServiceType        : MySnapIn
Version            : 1

Address            : http://MyServer.com:80/Citrix/MyContract/PeerAPI/v1
Binding            : wcf_HTTP_kerb
InterfaceType      : Peer
Metadata           : {}
MetadataMap        : {}
ServiceAccount     : ENG\MyAccount$
ServiceAccountSid  : S-1-5-21-1155438255-2213498043-2452000591-1104
ServiceGroupName   : MyService
ServiceGroupUid    : 2b990d5a-bba9-413b-aa08-e104e67f89bc
ServiceInstanceUid : 8f822ed6-42f3-4a26-911a-a4a6a87c0ef2
ServiceType        : MySnapIn
Version            : 1

Address            : http://MyServer.com:80/Citrix/MyContract/MyServiceEnvTestAPI/v1
Binding            : wcf_HTTP_kerb
InterfaceType      : EnvironmentTest
Metadata           : {}
MetadataMap        : {}
ServiceAccount     : ENG\MyAccount$
ServiceAccountSid  : S-1-5-21-1155438255-2213498043-2452000591-1104
ServiceGroupName   : MyService
ServiceGroupUid    : 2b990d5a-bba9-413b-aa08-e104e67f89bc
ServiceInstanceUid : d2d40d9b-2a5d-4c5a-b9ca-a7f73cfe4f2
ServiceType        : MySnapIn
Version : 1

Address : http://MyServer.com:80/Citrix/MyContract/MyServiceAPI/v1
Binding  : wcf_HTTP_kerb
InterfaceType : InterService
Metadata : {}
MetadataMap : {}
ServiceAccount : ENG\MyAccount$
ServiceAccountSid : S-1-5-21-1155438255-2213498043-2452000591-1104
ServiceGroupName : MyService
ServiceGroupUid : 2b990d5a-bba9-413b-aa08-e104e67f89bc
ServiceInstanceUid : 5d428970-2ba1-4336-b8d0-f3aa961b8983
ServiceType : MySnapIn
Version : 1

Return all the service instances that are registered in the Configuration Service.

-------------------------- EXAMPLE 2 --------------------------

C:\>Get-ConfigRegisteredServiceInstance -InterfaceType "SDK"

Address : http://MyServer.com:80/Citrix/MyContract/v1
Binding  : wcf_HTTP_kerb
InterfaceType : SDK
Metadata : {}
MetadataMap : {}
ServiceAccount : ENG\MyAccount$
ServiceAccountSid : S-1-5-21-1155438255-2213498043-2452000591-1104
ServiceGroupName : MyService
ServiceGroupUid : 2b990d5a-bba9-413b-aa08-e104e67f89bc
ServiceInstanceUid : 8dc38b5a-3fbb-457c-b326-6c41c94c18d5
ServiceType : MySnapIn
Version : 1

Return all the service instances that are registered in the Configuration Service and are of type 'SDK'.
Get-ConfigService

Oct 10, 2013
Gets the service record entries for the Configuration Service.

Syntax

Detailed Description
Returns instances of the Configuration Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Related topics
Parameters

-Property <String[]>
Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-ReturnTotalRecordCount <SwitchParameter>
When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Config_Filtering for details.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

-MaxRecordCount <Int32>
Specifies the maximum number of records to return.

<p>| Required? | false |</p>
<table>
<thead>
<tr>
<th>Default Value</th>
<th>250</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Skip<Int32>**

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | false |

**-SortBy<String>**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |
| Accept Pipeline Input? | false |

**-Filter<String>**

Gets records that match a PowerShell-style filter expression. See about_Config_Filtering for details.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values
The Get-ConfigServiceInstance command returns an object containing the following properties.

**Uid <Integer>**
Specifies the unique identifier for the service in the group. The unique identifier is an index number.

**ServiceHostId <Guid>**
Specifies the unique identifier for the service instance.

**DNSName <String>**
Specifies the domain name of the host on which the service runs.

**MachineName <String>**
Specifies the short name of the host on which the service runs.

**CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>**
Specifies whether the service is running, started but inactive, stopped, or failed.

**LastStartTime <DateTime>**
Specifies the date and time at which the service was last restarted.

**LastActivityTime <DateTime>**
Specifies the date and time at which the service was last stopped or restarted.

**OSType**
Specifies the operating system installed on the host on which the service runs.

**OSVersion**
Specifies the version of the operating system installed on the host on which the service runs.

**ServiceVersion**
Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

**DatabaseUserName <string>**
Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

**Sid <string>**
Specifies the Active Directory account security identifier for the machine on which the service instance is running.

**ActiveSiteServices <string[]>**
Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.
Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ConfigService

Uid : 1
ServiceHostId : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName : MyServer.company.com
MachineName : MYSERVER
CurrentState : On
LastStartTime : 04/04/2011 15:25:38
LastActivityTime : 04/04/2011 15:33:39
OSType : Win32NT
OSVersion : 6.1.7600.0
ServiceVersion : 5.1.0.0
DatabaseUserName : NT AUTHORITY\NETWORK SERVICE
SID : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {MySiteService1, MySiteService2...}

Get all the instances of the Configuration Service running in the current service group.
Get-ConfigServiceAddedCapability

Oct 10, 2013
Gets any added capabilities for the Configuration Service on the controller.

Syntax

Get-ConfigServiceAddedCapability [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Enables updates to the Configuration Service on the controller to be detected.

You do not need to configure a database connection before using this command.

Related topics

Parameters

- **-AdminAddress**<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ConfigServiceAddedCapability
Get the added capabilities of the Configuration Service.
Get-ConfigServiceGroup

Oct 10, 2013
Gets the service groups that match the parameters supplied.

Syntax

Detailed Description
Use this cmdlet to retrieve existing service groups that match the parameters supplied. If no parameters are supplied, all the service groups are returned.

Related topics

Parameters

(ServiceGroupUid)<Guid>
The unique identifier for the service group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

(ServiceGroupName)<String>
The service type for the service group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

(ServiceType)<String>
The service type for the service group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

(ReturnTotalRecordCount)<SwitchParameter>
See about_Configfiltering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**MaxRecordCount**<Int32>

See about_Acctfiltering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Skip**<Int32>

See about_Acctfiltering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**SortBy**<String>

See about_Acctfiltering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Filter**<String>

See about_Acctfiltering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>
Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**

Citrix.Configuration.Sdk.ServiceGroup

This represents a service instance and has the following parameters;

- ServiceGroupUid <Guid>
The unique identifier for the service group.
- ServiceGroupName <string>
The name of the service group.
- ServiceType <string>
The type of the service group.
- Metadata <Citrix.Configuration.Sdk.Metadata[]>
The metadata for the service group.

**Notes**

In the case of failure, the following errors can result.

**Error Codes**

- DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.
- PartialData Only a subset of the available data was returned.
- CouldNotQueryDatabase The Query required to get the database was not defined.
- CommunicationError An error occurred while communicating with the service.
- InvalidFilter A filtering expression was supplied that could not be interpreted for this cmdlet.
- ExceptionThrown An unexpected error occurred.

To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\>Get-ConfigServiceGroup
Return all the service groups that are registered in the Configuration Service.

-------------------------- EXAMPLE 2 --------------------------

C:\>Get-ConfigRegisteredServiceInstance -ServiceType "config"
Return all the service groups that are registered in the Configuration Service and are of type 'config' (i.e. the service groups
for the Configuration Service).
Get-ConfigServiceInstance

Oct 10, 2013

Gets the service instance entries for the Configuration Service.

Syntax

Get-ConfigServiceInstance [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns service interfaces published by the instance of the Configuration Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Related topics

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Return Values

Citrix.Configuration.Sdk.ServiceInstance

The Get-ConfigServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.
ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Config.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations
InterService - for operations between different services
Peer - for communications between services of the same type

Metadata <Citrix.Configuration.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError
An error occurred in the service while attempting a database operation.

**DatabaseNotConfigured**

The operation could not be completed because the database for the service is not configured.

**DataStoreException**

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

**PermissionDenied**

You do not have permission to execute this command.

**AuthorizationError**

There was a problem communicating with the Citrix Delegated Administration Service.

**CommunicationError**

There was a problem communicating with the remote service.

**ExceptionThrown**

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ConfigServiceInstance

<table>
<thead>
<tr>
<th>Address</th>
<th><a href="http://MyServer.com:80/Citrix/ConfigurationService">http://MyServer.com:80/Citrix/ConfigurationService</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Binding</td>
<td>wcf_HTTP_kerb</td>
</tr>
<tr>
<td>InterfaceType</td>
<td>SDK</td>
</tr>
<tr>
<td>Metadata</td>
<td></td>
</tr>
<tr>
<td>MetadataMap</td>
<td></td>
</tr>
<tr>
<td>ServiceAccount</td>
<td>ENG\MyAccount$</td>
</tr>
<tr>
<td>ServiceAccountSid</td>
<td>S-1-5-21-2406005612-3133289213-1653143164</td>
</tr>
<tr>
<td>ServiceGroupName</td>
<td>MyServiceGroup</td>
</tr>
<tr>
<td>ServiceGroupUid</td>
<td>a88d2f6b-c00a-4f76-9c38-e8330093b54d</td>
</tr>
<tr>
<td>ServiceInstanceId</td>
<td>0000000000-0000-0000-0000-000000000000</td>
</tr>
<tr>
<td>ServiceType</td>
<td>Config</td>
</tr>
<tr>
<td>Version</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Address</th>
<th><a href="http://MyServer.com:80/Citrix/ConfigurationService/IServiceApi">http://MyServer.com:80/Citrix/ConfigurationService/IServiceApi</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Binding</td>
<td>wcf_HTTP_kerb</td>
</tr>
<tr>
<td>InterfaceType</td>
<td>InterService</td>
</tr>
<tr>
<td>Metadata</td>
<td></td>
</tr>
<tr>
<td>MetadataMap</td>
<td></td>
</tr>
<tr>
<td>ServiceAccount</td>
<td>ENG\MyAccount</td>
</tr>
<tr>
<td>ServiceAccountSid</td>
<td>S-1-5-21-2406005612-3133289213-1653143164</td>
</tr>
</tbody>
</table>
Get all instances of the Configuration Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.
Get-ConfigServiceStatus

Oct 10, 2013
Gets the current status of the Configuration Service on the controller.

Syntax
Get-ConfigServiceStatus [-AdminAddress <String>] [CommonParameters]

Detailed Description
Enables the status of the Configuration Service on the controller to be monitored. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured.

Related topics
Set-ConfigDBConnection
Test-ConfigDBConnection
Get-ConfigDBConnection
Get-ConfigDBSchema

Parameters
-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values

The Get-ConfigServiceStatus command returns an object containing the status of the Configuration Service together with extra diagnostics information.

DBUnconfigured
The Configuration Service does not have a database connection configured.

DBRejectedConnection
The database rejected the logon attempt from the Configuration Service. This may be because the service attempted to
log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the Configuration Service schema has not been added to the database.

DBNot Found

The specified database could not be located with the configured connection string.

DBNewerVersionThanService

The version of the Configuration Service currently in use is incompatible with the version of the Configuration Service schema on the database. Upgrade the Configuration Service to a more recent version.

DBOlderVersionThanService

The version of the Configuration Service schema on the database is incompatible with the version of the Configuration Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Configuration Service is running and is connected to a database containing a valid schema.

Failed

The Configuration Service has failed.

Unknown

(0) The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ConfigServiceStatus

DBUnconfigured
Get the current status of the Configuration Service.
Get-ConfigSite

Oct 10, 2013
Gets the site.

Syntax

Get-ConfigSite [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Get-ConfigSite cmdlet gets the site.

A XenDesktop installation has only a single site instance.

Related topics

Set-ConfigSite
Import-ConfigFeatureTable

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

Return Values

Get-ConfigSite returns the site instance.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-ConfigSite
Gets the site.
Import-ConfigFeatureTable

Oct 10, 2013
Sets the feature table of the site.

Syntax

Import-ConfigFeatureTable [-Path] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Import-ConfigFeatureTable -Content <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Related topics

Export-ConfigFeatureTable
Get-ConfigSite
Set-ConfigSite
Get-ConfigProduct
Get-ConfigProductEdition
Get-ConfigProductFeature
Get-ConfigProductVersion
Get-ConfigLicensingModel

Parameters

-Path <String>
The path to the file containing the feature table

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Content <String>
Set the site's feature table.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>
**-LoggingId<Guid>**

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Examples**

------------- EXAMPLE 1 -------------

C:\PS> Import-ConfigFeatureTable $xml

Specifies the use of a Platinum edition license. A suitable license must be available on the site's license server.
Register-ConfigServiceInstance

Oct 10, 2013

Allows the registration of a service instance.

Syntax

Register-ConfigServiceInstance -ServiceGroupUid <Guid> -ServiceGroupName <String> -ServiceType <String> -Address <String> -Binding <String> -Version <Int32> -ServiceAccount <String> -InterfaceType <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Register-ConfigServiceInstance -ServiceGroupUid <Guid> -ServiceGroupName <String> -ServiceType <String> -Address <String> -Binding <String> -Version <Int32> -ServiceAccountSid <String> -InterfaceType <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Register-ConfigServiceInstance -ServiceInstance <ServiceInstance[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Use this cmdlet to register service instance items in the Configuration Service. Service instances can be registered either by retrieving the data directly from other services or by manually entering the details into this command.

If the service group specified by the service instance already exists, the service is added to the service group, otherwise a new service group is created to hold the service instance.

Related topics

Unregister-ConfigRegisteredServiceInstance
Add-ConfigRegisteredServiceInstanceMetadata
Set-ConfigRegisteredServiceInstanceMetadata
Remove-ConfigRegisteredServiceInstanceMetadata
Add-ConfigServiceGroupMetadata
Set-ConfigServiceGroupMetadata
Remove-ConfigServiceGroupMetadata

Parameters

-ServiceGroupUid <Guid>

The Service Group Unique Identifier

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
-ServiceGroupName<String>

The name of the service group

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-ServiceType<String>

The type of the service group

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-Address<String>

The address that is used to access the service instance.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-Binding<String>

The binding type that must be used to access the service instance.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-Version<Int32>

The version of the service instance.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-ServiceAccount<String>
The AD computer account name (domain qualified) for the computer on which the service instance is hosted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **InterfaceType**<String>

The type of interface that the service provides (i.e. SDK, InterService, Peer).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **ServiceAccountSid**<String>

The AD computer account Sid for the computer on which the service instance is hosted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **ServiceInstance**<ServiceInstance[]>

The service instances to register.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

Specifies the logging id of the high-level operation that this cmdlet is part of.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can be provide this as a host
name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.Configuration.Sdk.ServiceInstance An object with the following parameters can be used to register a service instance. Address, ServiceGroupUid, ServiceGroupName, ServiceType, Binding, Version, InterfaceType.

**Return Values**

Citrix.Configuration.Sdk.ServiceInstance

This represents a service instance and has the following parameters;

- **ServiceGroupUid <Guid>**
  The unique identifier for the service group to which the service instance belongs.

- **ServiceGroupName <string>**
  The name of the service group to which the service instance belongs.

- **ServiceInstanceUid <Guid>**
  The unique identifier for the service instance.

- **ServiceType <string>**
  The type of the service group.

- **Address <string>**
  The contact address for the service instance.

- **Binding <string>**
  The binding to use for connections to the service instance.

- **Version <int>**
  The version of the service instance.

- **ServiceAccount <string>**
  The AD computer account for the computer that is providing the service instance.

- **ServiceAccountSid <string>**
  The AD computer account SID for the computer that is providing the service instance.

- **InterfaceType <string>**
The interface type for the service instance.

Metadata `<Citrix.Configuration.Sdk.Metadata[]>`

The metadata for the service instance.

Notes

In the case of failure, the following errors can result:

Error Codes

- **ActiveDirectoryAccountResolutionFailed** The account name provided could not be found in Active Directory.
- **ServiceGroupWithSameUidExistsForDifferentGroupName** The service group name or service type do not match the service group found with the specified uid.
- **ServiceTypeAlreadyExists** A different service group with the same type is registered already in the Configuration Service.
- **DatabaseError** An error occurred in the service while attempting a database operation.
- **DatabaseNotConfigured** The operation could not be completed because the database for the service is not configured.
- **DataStoreException** An error occurred in the service while attempting a database operation – communication with the database failed for various reasons.
- **CommunicationError** An error occurred while communicating with the service.
- **ExceptionThrown** An unexpected error occurred.

For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Get-ConfigServiceInstance | Register-ConfigServiceInstance

Gets the service instances for the Configuration Service and registers them.
Remove-ConfigRegisteredServiceInstanceMetadata

Oct 10, 2013
Removes metadata from the given ServiceInstance.

Syntax

Remove-ConfigRegisteredServiceInstanceMetadata [-ServiceInstanceUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-ConfigRegisteredServiceInstanceMetadata [-ServiceInstanceUid] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-ConfigRegisteredServiceInstanceMetadata [-InputObject] <ServiceInstance[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-ConfigRegisteredServiceInstanceMetadata [-InputObject] <ServiceInstance[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given ServiceInstance.

Related topics

Add-ConfigRegisteredServiceInstanceMetadata
Set-ConfigRegisteredServiceInstanceMetadata

Parameters

- **-ServiceInstanceUid** <Guid>

Id of the ServiceInstance

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-InputObject** <ServiceInstance[]>

Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>
### -Map<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with `new-object System.Collections.Generic.Dictionary[String,String]`).

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue) |

### -Name<String>

The metadata property to remove.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | false |

### -LoggingId<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

### -AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

### Notes

If the command fails, the following errors can be returned.

### Error Codes
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

------------- EXAMPLE 1 -------------
c:\PS>Get-ConfigServiceInstance | Remove-ConfigRegisteredServiceInstanceMetadata
Remove all metadata from all ServiceInstance objects.
Remove-ConfigServiceGroup

Oct 10, 2013
Removes service groups.

Syntax


Detailed Description

Use this cmdlet to remove a service group and all the service instances that it contains.

Related topics

Register-ConfigServiceInstance
Add-ConfigServiceGroupMetadata
Set-ConfigServiceGroupMetadata
Remove-ConfigServiceGroupMetadata

Parameters

-ServiceGroupUid <Guid>

The unique identifier for the service group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the logging id of the high-level operation this cmdlet invocation is part of.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a hostname or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>----------</td>
<td>-------</td>
</tr>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

In the case of failure, the following errors can result.

Error Codes  
ServiceGroupObjectNotFound The service group specified could not be located.  
DatabaseError An error occurred in the service while attempting a database operation.  
DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.  
DataStoreException An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.  
CommunicationError An error occurred while communicating with the service.  
ExceptionThrown An unexpected error occurred.

For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Remove-ConfigServiceGroup -ServiceGroupUid 31951f6f-7703-4abb-938a-2861d76ecfea  
Removes the service group (and all contained service instances) for the service group with a unique identifier of '31951f6f-7703-4abb-938a-2861d76ecfea'.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-configServiceGroup | remove-ConfigServiceGroup  
Removes all the service groups (and all contained service instances) for the XenDesktop deployment.
Remove-ConfigServiceGroupMetadata

Oct 10, 2013
Removes metadata from the given ServiceGroup.

Syntax

Remove-ConfigServiceGroupMetadata [-ServiceGroupUid] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-ConfigServiceGroupMetadata [-ServiceGroupUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-ConfigServiceGroupMetadata [-InputObject] <ServiceGroup[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-ConfigServiceGroupMetadata [-InputObject] <ServiceGroup[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given ServiceGroup.

Related topics

Add-ConfigServiceGroupMetadata
Set-ConfigServiceGroupMetadata

Parameters

-ServiceGroupUid <Guid>

Id of the ServiceGroup

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-InputObject <ServiceGroup[]>

Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>
- **Name**: String

  The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Map**: PSObject

  Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with `new-object "System.Collections.Generic.Dictionary[String,String]"`). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **LoggingId**: Guid

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**: String

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

---------------------------- EXAMPLE 1 ----------------------------

c:\PS>Get-ConfigServiceGroup | % { Remove-ConfigServiceGroupMetadata -Map $_.MetadataMap }
Remove all metadata from all ServiceGroup objects.
Remove-ConfigServiceMetadata

Oct 10, 2013
Removes metadata from the given Service.

Syntax

Remove-ConfigServiceMetadata [-ServiceHostId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters]

Remove-ConfigServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters]


Remove-ConfigServiceMetadata [-InputObject] <Service[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters]

Detailed Description

Provides the ability to remove metadata from the given Service.

Related topics

Set-ConfigServiceMetadata

Parameters

-**ServiceHostId** <Guid>

Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
</tr>
</thead>
</table>

| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-**InputObject** <Service[]>

Objects to which metadata is to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
</tr>
</thead>
</table>

| Accept Pipeline Input? | true (ByValue) |

-**Name** <String>
The metadata property to remove.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

- `Map<PSObject>`

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with `new-object "System.Collections.Generic.Dictionary[String,String]"`). The properties whose names match keys in the map will be removed.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

- `LoggingId<Guid>`

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- `AdminAddress<String>`

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Notes

If the command fails, the following errors can be returned.

Error Codes

---------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-ConfigService | % { Remove-ConfigServiceMetadata -Map $_.MetadataMap }
Remove all metadata from all Service objects.
Remove-ConfigSiteMetadata

Oct 10, 2013
Removes metadata from the given Site.

Syntax

Remove-ConfigSiteMetadata -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-ConfigSiteMetadata -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given Site.

Related topics

Set-ConfigSiteMetadata

Parameters

- **Name**<String>

The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Map**<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
**-AdminAddress**: <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

----------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ConfigSite | % { Remove-ConfigSiteMetadata -Map $_.MetadataMap }
Remove all metadata from all Site objects.
Reset-ConfigServiceGroupMembership

Oct 10, 2013
Reloads the access permissions and configuration service locations for the Configuration Service.

Syntax

Reset-ConfigServiceGroupMembership [-ConfigServiceInstance] <ServiceInstance[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Enables you to reload Configuration Service access permissions and configuration service locations. The Reset-ConfigServiceGroupMembership command must be run on at least one instance of the service type (Config) after installation and registration with the configuration service. Without this operation, the Configuration services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-ConfigServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Related topics

Parameters

- **-ConfigServiceInstance** <ServiceInstance[]>

  Specifies the configuration service instance object that represents the service instance for the type 'InterService' that references a configuration service for the deployment.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress** <String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>
Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-ConfigServiceGroupMembership command.

Notes

If the command fails, the following errors can be returned.

Error Codes

-----------

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-ConfigServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config -AdminAddress OtherServer.example.com | Reset-ConfigServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.
Set-ConfigDBConnection

Oct 10, 2013
Configures a database connection for the Configuration Service.

Syntax


Detailed Description

Configures a connection to a database in which the Configuration Service can store its state. The service will attempt to connect and start using the database immediately after the connection is configured. The database connection string is updated to the specified value regardless of whether it is valid or not. Specifying an invalid connection string prevents a service from functioning until the error is corrected.

After a connection is configured, you cannot alter it without first clearing it (by setting the connection to $null).

You do not need to configure a database connection to use this command.

Related topics

Get-ConfigServiceStatus
Get-ConfigDBConnection
Test-ConfigDBConnection

Parameters

-DBConnection<String>

Specifies the database connection string to be used by the Configuration Service. Passing in $null will clear any existing database connection configured.

| Required?  | true |
| Default Value | |
| Accept Pipeline Input? | false |

-Force<SwitchParameter>

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

| Required?  | false |
| Default Value | |
| Accept Pipeline Input? | false |

-LoggingId<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and StopLogHighLevelOperation cmdlets.

| Required?  | false |
| Default Value | |

- **AdminAddress**:<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>false</th>
</tr>
</thead>
</table>

Return Values


The Set-ConfigDBConnection command returns an object containing the status of the Configuration Service together with extra diagnostics information.

**DBUnconfigured**

The Configuration Service does not have a database connection configured.

**DBRejectedConnection**

The database rejected the logon attempt from the Configuration Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**

The expected stored procedures are missing from the database. This may be because the Configuration Service schema has not been added to the database.

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**

The version of the Configuration Service currently in use is incompatible with the version of the Configuration Service schema on the database. Upgrade the Configuration Service to a more recent version.

**DBOlderVersionThanService**

The version of the Configuration Service schema on the database is incompatible with the version of the Configuration Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**

A database schema upgrade is currently in progress.

**OK**

The Configuration Service is running and is connected to a database containing a valid schema.

**Failed**

The Configuration Service has failed.

**Unknown**

The status of the Configuration Service cannot be determined.
Notes
If the command fails, the following errors can be returned.

Error Codes
-----------
InvalidDBConnectionString
The database connection string has an invalid format.
DatabaseConnectionDetailsAlreadyConfigured
There was already a database connection configured. After a configuration is set, it can only be set to $null.
PermissionDenied
You do not have permission to execute this command.
AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.
CommunicationError
There was a problem communicating with the remote service.
ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Set-ConfigDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Configures a database connection string for the Configuration Service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Set-ConfigDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Configures an invalid database connection string for the Configuration Service.
Set-ConfigRegisteredServiceInstance

Oct 10, 2013
Updates a service instance.

Syntax

Set-ConfigRegisteredServiceInstance -ServiceInstanceUid <Guid> -Address <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Use this cmdlet to change the address property of an existing service instance that is registered in the Configuration Service.

Related topics

Get-ConfigRegisteredServiceInstance
Register-ConfigServiceInstance
Unregister-ConfigRegisteredServiceInstance

Parameters

-ServiceInstanceUid <Guid>
The unique identifier for the service instance to be updated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-Address <String>
The new address for the service instance.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-PassThru <SwitchParameter>
Defines whether or not the command returns a result showing the new state of the updated service instance.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

-LoggingId <Guid>
Specifies the logging id of the high-level operation this cmdlet invocation is part of.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.Configuration.Sdk.ServiceInstance

This represents a service instance and has the following parameters:

ServiceGroupUid <Guid>
The unique identifier for the service group to which the service instance belongs.

ServiceGroupName <string>
The name of the service group to which the service instance belongs.

ServiceInstanceUid <Guid>
The unique identifier for the service instance.

ServiceType <string>
The type of the service group.

Address <string>
The contact address for the service instance.

Binding <string>
The binding to use for connections to the service instance.

Version <int>
The version of the service instance.

ServiceAccount <string>
The AD computer account for the computer that is providing the service instance.

ServiceAccountSid <string>
The AD computer account SID for the computer that is providing the service instance.

InterfaceType <string>
The interface type for the service instance.

Metadata <Citrix.Configuration.Sdk.Metadata[]>
The metadata for the service instance.

Notes

In the case of failure, the following errors can result.

Error Codes

ObjectToUpdateDoesNotExist The service instance specified could not be located.
DatabaseError An error occurred in the service while attempting a database operation.
DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.
DataStoreException An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.
CommunicationError An error occurred while communicating with the service.
ExceptionThrown An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Set-ConfigResgisteredService -ServiceInstanceUid "9805f39d-99eb-44f0-8f63-9d8e3f1228e0" -Address "http://myServer.com/Citrix/sdkHostingUnitService"
Update the service instance with the unique identifier of '9805f39d-99eb-44f0-8f63-9d8e3f1228e0' to use the new address.
**Set-ConfigRegisteredServiceInstanceMetadata**

Oct 10, 2013

Adds or updates metadata on the given ServiceInstance.

**Syntax**

Set-ConfigRegisteredServiceInstanceMetadata [-ServiceInstanceUid] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters]


Set-ConfigRegisteredServiceInstanceMetadata [-InputObject] <ServiceInstance[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters]


**Detailed Description**

Provides the ability for additional custom data to be stored against given ServiceInstance objects.

**Related topics**

Add-ConfigRegisteredServiceInstanceMetadata

Remove-ConfigRegisteredServiceInstanceMetadata

**Parameters**

- **-ServiceInstanceUid** `<Guid>`
  
  Id of the ServiceInstance

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-InputObject** `<ServiceInstance[]>`
  
  Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Name** `<String>`
  
  Specifies the property name of the metadata to be added. The property must be unique for the ServiceInstance specified. The property cannot contain any of the following characters `\/:*?"<>|[]()"`

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1647
-Value <String>
Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Map <PSObject>
Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
Set-ConfigRegisteredServiceInstanceMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>
Specifies the name of the property.

Value <string>
Specifies the value for the property.

Notes
If the command fails, the following errors can be returned.

Error Codes
InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> Set-ConfigRegisteredServiceInstanceMetadata -ServiceInstanceUid 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the ServiceInstance with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Set-ConfigServiceGroupMetadata

Oct 10, 2013

Adds or updates metadata on the given ServiceGroup.

Syntax

Set-ConfigServiceGroupMetadata [-ServiceGroupUid] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-ConfigServiceGroupMetadata [-ServiceGroupUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-ConfigServiceGroupMetadata [-InputObject] <ServiceGroup[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-ConfigServiceGroupMetadata [-InputObject] <ServiceGroup[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability for additional custom data to be stored against given ServiceGroup objects.

Related topics

Add-ConfigServiceGroupMetadata
Remove-ConfigServiceGroupMetadata

Parameters

-ServiceGroupUid <Guid>

Id of the ServiceGroup

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-InputObject <ServiceGroup[]>

Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name <String>

Specifies the property name of the metadata to be added. The property must be unique for the ServiceGroup specified. The property cannot contain any of the following characters \/:.*=?<> |[]"
### -Value<String>

Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

### -Map<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with `new-object "System.Collections.Generic.Dictionary[String,String]"`).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

### -LoggingId<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the `Start-LogHighLevelOperation` and `Stop-LogHighLevelOperation` cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

### -AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

## Return Values

```
```

`Set-ConfigServiceGroupMetadata` returns a dictionary containing the new (name, value)-pairs.
Key <string>
Specifies the name of the property.

Value <string>
Specifies the value for the property.

Notes
If the command fails, the following errors can be returned.

Error Codes
-------------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Set-ConfigServiceGroupMetadata -ServiceGroupUid 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value
<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the ServiceGroup with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Set-ConfigServiceMetadata

Oct 10, 2013
Adds or updates metadata on the given Service.

Syntax

Set-ConfigServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-ConfigServiceMetadata [-ServiceHostId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-ConfigServiceMetadata [-InputObject] <Service[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-ConfigServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Allows you to store additional custom data against given Service objects.

Related topics

Remove-ConfigServiceMetadata

Parameters

-ServiceHostId <Guid>

Id of the Service

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-InputObject <Service[]>

Objects to which metadata is to be added.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue) |

-Map <PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @"name1" = "val1";
"name2" = "val2") or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name**<String>

  Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/,:#.*?=<>|[]()"

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Value**<String>

  Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values


Set-ConfigServiceMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>
Specifies the name of the property.

Value <string>
Specifies the value for the property.

Notes
If the command fails, the following errors can be returned.

Error Codes

----------

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Set-ConfigServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2A06DD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DD0805C'.
**Set-ConfigSite**

Oct 10, 2013
Changes the overall settings of the site.

Syntax

```
```

Detailed Description

The Set-ConfigSite cmdlet modifies properties of the site.

The site is a top-level logical representation of the XenDesktop site, from the perspective of the configuration services running within the site.

A XenDesktop installation has only a single site instance.

Modifications to the product code, product edition, product version and licensing model properties are successful only if their values are consistent with the feature table. Use the Get-ConfigProduct, Get-ConfigProductEdition, Get-ConfigProductVersion and Get-ConfigLicensingModel cmdlets to determine consistent values.

To configure the site, first import the feature table using the Import-ConfigFeatureTable cmdlet.

Related topics

- Export-ConfigFeatureTable
- Get-ConfigSite
- Get-ConfigProduct
- Get-ConfigProductEdition
- Get-ConfigProductFeature
- Get-ConfigProductVersion
- Get-ConfigLicensingModel

Parameters

- **-SiteName <String>**

Changes the name of the site.

<table>
<thead>
<tr>
<th>Required?</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>False</td>
</tr>
</tbody>
</table>
- **ProductCode**: `<String>`

Changes the product code.

The Get-ConfigProduct cmdlet returns a list of supported values.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ProductEdition**: `<String>`

Changes the license edition. A license matching the specified edition must be available within the site's license server.

The Get-ConfigProductEdition returns a list of supported values.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ProductVersion**: `<String>`

Changes the product version.

The Get-ConfigProductVersion returns a list of supported values.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LicensingModel**: `<String>`

Changes the license model. A license matching the specified model must be available within the site's license server.

The Get-ConfigLicensingModel returns a list of supported values.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LicenseServerName**: `<String>`
Changes the machine used by the brokering services to obtain licenses for desktop and application session brokering. The specified machine must be running a Citrix license server and have suitable licenses installed.

The license server machine can be specified by its DNS name ('machine.domain') or its numeric IP address.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>-LicenseServerPort&lt;Int32&gt;</td>
<td>Changes the port number on the license server machine used by the brokering services to contact the Citrix license server.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>-LoggingId&lt;Guid&gt;</td>
<td>Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>-AdminAddress&lt;String&gt;</td>
<td>Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Site

Examples
C:\PS> Set-ConfigSite -ProductEdition PLT
Specifies the use of a Platinum edition license. A suitable license must be available on the site's license server.
Set-ConfigSiteMetadata

Oct 10, 2013
Adds or updates metadata on the Site.

Syntax

Set-ConfigSiteMetadata -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Set-ConfigSiteMetadata -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

Provides the ability for additional custom data to be stored against the Site.

Related topics

Remove-ConfigSiteMetadata

Parameters

-Name <String>

Specifies the property name of the metadata to be added. The property must be unique for the Site specified. The property cannot contain any of the following characters \/:#.*?=<{}[]"'

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-Value <String>

Specifies the value for the property.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-Map <PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | |

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1662
- **LoggingId** `<Guid>`

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress** `<String>`

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**


Set-ConfigSiteMetadata returns a dictionary containing the new (name, value)-pairs.

**Key** `<string>`

Specifies the name of the property.

**Value** `<string>`

Specifies the value for the property.

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

--------

InvalidParameterCombination

The cmdlet parameters are inconsistent.
UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> Set-ConfigSiteMetadata -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Site.
Test-ConfigDBConnection

Oct 10, 2013
Tests a database connection for the Configuration Service.

Syntax

Test-ConfigDBConnection [-DBConnection] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Tests a connection to the database in which the Configuration Service can store its state. The service will attempt to connect to the database without affecting the current connection to the database.

You do not have to clear the connection to use this command.

Related topics

Get-ConfigServiceStatus
Get-ConfigDBConnection
Set-ConfigDBConnection

Parameters

- **DBConnection**<String>

  Specifies the database connection string to be tested by the Configuration Service.

<table>
<thead>
<tr>
<th>Required?</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

  Default Value: Localhost. Once a value is provided by any cmdlet, this value becomes the default.

Return Values

The Test-ConfigDBConnection command returns an object containing the status of the Configuration Service if the connection string of the specified data store were to be set to the string being tested, together with extra diagnostics information for the specified connection string.

DBRejectedConnection

The database rejected the logon attempt from the Configuration Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the Configuration Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBNewerVersionThanService

The version of the Configuration Service currently in use is incompatible with the version of the Configuration Service schema on the database. Upgrade the Configuration Service to a more recent version.

DBOlderVersionThanService

The version of the Configuration Service schema on the database is incompatible with the version of the Configuration Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Set-ConfigDBConnection command would succeed if it were executed with the supplied connection string.

Failed

The Configuration Service has failed.

Unknown

The status of the Configuration Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

-------------

InvalidDBCConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

https://docs.citrix.com

© 1999-2017 Citrix Systems, Inc. All rights reserved.

p.1666
The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Test-ConfigDBConnection -DBConnection "Server=servername\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Tests a database connection string for the Configuration Service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Test-ConfigDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Tests an invalid database connection string for the Configuration Service.
Test-ConfigServiceInstanceAvailability

Oct 10, 2013
Tests whether the supplied service instances are responding to requests.

Syntax

Test-ConfigServiceInstanceAvailability [-ServiceInstance <ServiceInstance[]>] [-MaxDelaySeconds <Int32>] [-ForceWaitForOneOfEachType] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Related topics

Register-ConfigServiceInstance
Unregister-ConfigRegisteredServiceInstance
Get-ConfigRegisteredServiceInstance

Parameters

(ServiceInstance)<ServiceInstance[]>
The service instances to test.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue) |

(MaxDelaySeconds)<Int32>
The timeout period to wait before concluding that services are unresponsive.

| Required? | false |
| Default Value | Infinite |
| Accept Pipeline Input? | false |

(ForceWaitForOneOfEachType)<SwitchParameter>
If at least one of each type of service responds, finish immediately.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |
- **AdminAddress**<String>

  Specifies the host name or IP address of the controller to which the PowerShell snap-in connects.

  Required? | false
  Default Value | 'LocalHost'. Once a value is specified by any command, this value becomes the new default.
  Accept Pipeline Input? | false

Return Values

System.Management.Automation.PSObject

This represents a service instance and has the following parameters:

  ServiceGroupUid <Guid>
  The unique identifier for the service group to which the service instance belongs.

  ServiceGroupName <string>
  The name of the service group that the service instance is part of.

  ServiceInstanceUid <Guid>
  The unique identifier for the service instance.

  ServiceType <string>
  The type of the service group.

  Address <string>
  The contact address for the service instance.

  Binding <string>
  The binding to use for connections to the service instance.

  Version <int>
  The version of the service instance.

  ServiceAccount <string>
  The AD computer account for the computer that is providing the service instance.

  ServiceAccountSid <string>
  The AD computer account SID for the computer that is providing the service instance.

  InterfaceType <string>
The interface type for the service instance.

Metadata <Citrix.Configuration.Sdk.Metadata[]>

The metadata for the service instance.


An enumeration value indicating whether the service is Responding, NotResponding, Unknown, or BadBindingType.

ResponseTime <System.TimeSpan>

The interval elapsed between hailing the service and getting a definite response

Notes

The Availability Status Codes are:
- Responding: Got a positive response
- NotResponding: Got a response, but it was negative or the connection was refused
- Unknown: Did not respond in time / timed-out
- BadBindingType: Binding parameter in ServiceInstance is not wcf_HTTP_kerb

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\> Get-ConfigRegisteredServiceInstance | Test-ConfigServiceInstanceAvailability -ForceWaitForOneOfEachType
Test all the service instances that are registered in the Configuration Service, returning when one of each type is responding.

-------------------------- EXAMPLE 2 --------------------------

C:\> Test-ConfigServiceInstanceAvailability -ServiceInstance $services -MaxDelaySeconds 5
Test each of the given services, allowing a 5 second time-out.
Unregister-ConfigRegisteredServiceInstance

Oct 10, 2013
Removes a service instance from the Configuration Service registry.

Syntax


Detailed Description

Use this cmdlet to remove a service instance from the Configuration Service registry. This does not remove any service groups (if all service instances for a Service Group are removed, an empty service group remains and must be removed using the Remove-ConfigServiceGroup command).

Related topics

Register-ConfigServiceInstance

Parameters

-ServiceInstanceUid <Guid>

The unique identifier for the service instance to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the logging id of the high-level operation this cmdlet invocation is part of.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>
Input Type

Citrix.Configuration.Sdk.ServiceInstance An object with a parameter called 'ServiceInstanceUid' can be used to unregister service instances.

Notes

In the case of failure, the following errors can result.

Error Codes

- ServiceInstanceObjectNotFound The service instance specified could not be located.
- DatabaseError An error occurred in the service while attempting a database operation.
- DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.
- DataStoreException An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.
- CommunicationError An error occurred while communicating with the service.
- ExceptionThrown An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

EXAMPLE 1

C:\PS>Get-ConfigRegisteredServiceInstance -ServiceType "Config" | Unregister-ConfigRegisteredServiceInstance
Unregisters all the service instances that are for a service type of 'Config' from the Configuration Service instance register.
## Overview

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LogConfigurationLoggingSnapin</td>
<td>The Configuration Logging Service PowerShell snap-in provides administrative support.</td>
</tr>
<tr>
<td>Log Filtering</td>
<td>Describes the common filtering options for XenDesktop cmdlets.</td>
</tr>
<tr>
<td>logical operators</td>
<td>Describes the operators that connect statements in Windows PowerShell.</td>
</tr>
</tbody>
</table>

## Cmdlets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export-LogReportHtml</td>
<td>Exports Configuration Logging data into a HTML report.</td>
</tr>
<tr>
<td>Get-LogDataStore</td>
<td>Gets details for each of the ConfigurationLogging data stores.</td>
</tr>
<tr>
<td>Get-LogDBConnection</td>
<td>Gets the database string for the specified data store used by the ConfigurationLogging Service.</td>
</tr>
<tr>
<td>Get-LogDBSchema</td>
<td>Gets a script that creates the ConfigurationLogging Service database schema for the specified data store.</td>
</tr>
<tr>
<td>Get-LogDBVersionChangeScript</td>
<td>Gets a script that updates the ConfigurationLogging Service database schema.</td>
</tr>
<tr>
<td>Get-LogHighLevelOperation</td>
<td>Gets high level operations.</td>
</tr>
<tr>
<td>Get-LogInstalledDBVersion</td>
<td>Gets a list of all available database schema versions for the ConfigurationLogging Service.</td>
</tr>
<tr>
<td>Get-LogLowLevelOperation</td>
<td>Gets low level operations.</td>
</tr>
<tr>
<td>Get-LogService</td>
<td>Gets the service record entries for the ConfigurationLogging Service.</td>
</tr>
<tr>
<td>Get-LogServiceAddedCapability</td>
<td>Gets any added capabilities for the ConfigurationLogging Service on the controller.</td>
</tr>
<tr>
<td>Get-LogServiceInstance</td>
<td>Gets the service instance entries for the ConfigurationLogging Service.</td>
</tr>
<tr>
<td>Get-LogServiceStatus</td>
<td>Gets the current status of the ConfigurationLogging Service on the controller.</td>
</tr>
<tr>
<td>Get-LogSite</td>
<td>Gets global configuration logging settings.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Get-LogSummary</td>
<td>Gets operations logged within time intervals inside a date range.</td>
</tr>
<tr>
<td>Remove-LogOperation</td>
<td>Deletes configuration logs</td>
</tr>
<tr>
<td>Remove-LogServiceMetadata</td>
<td>Removes metadata from the given Service.</td>
</tr>
<tr>
<td>Remove-LogSiteMetadata</td>
<td>Removes metadata from the given Site.</td>
</tr>
<tr>
<td>Reset-LogDataStore</td>
<td>Refreshes the database string currently being used by the Log service.</td>
</tr>
<tr>
<td>Reset-LogServiceGroupMembership</td>
<td>Reloads the access permissions and configuration service locations for the ConfigurationLogging Service.</td>
</tr>
<tr>
<td>Set-LogDBConnection</td>
<td>Configures a database connection for the ConfigurationLogging Service.</td>
</tr>
<tr>
<td>Set-LogServiceMetadata</td>
<td>Adds or updates metadata on the given Service.</td>
</tr>
<tr>
<td>Set-LogSite</td>
<td>Sets global configuration logging settings.</td>
</tr>
<tr>
<td>Set-LogSiteMetadata</td>
<td>Adds or updates metadata on the given Site.</td>
</tr>
<tr>
<td>Start-LogHighLevelOperation</td>
<td>Logs the start of a high level operation.</td>
</tr>
<tr>
<td>Stop-LogHighLevelOperation</td>
<td>Logs the completion of a previously started high level operation.</td>
</tr>
<tr>
<td>Test-LogDBConnection</td>
<td>Tests a database connection for the ConfigurationLogging Service.</td>
</tr>
</tbody>
</table>
about_LogConfigurationLoggingSnapin

Oct 10, 2013

TOPIC

about_LogConfigurationLoggingSnapin

SHORT DESCRIPTION

The Configuration Logging Service PowerShell snap-in provides administrative functions for the Configuration Logging Service.

COMMAND PREFIX

All commands in this snap-in have the noun prefixed with 'Log'.

LONG DESCRIPTION

The Configuration Logging Service PowerShell snap-in enables both local and remote administration of the Configuration Logging Service.

The Configuration Logging Service logs configuration changes or administrator requested state changes made to the site. Configuration Logging can be configured, site wide, to be mandatory or optional. If mandatory logging is selected, then any attempts to change site configuration or state when the logging mechanism is unavailable are denied.

The Configuration Logging Service stores information about the logged changes in a database which can be configured to be separate from the site database.

The snap-in provides storage and configuration of these entities:

Site

The Configuration Logging Site object holds global settings which control the behaviour of the Configuration Logging Service. The site object can be configured by the Set-LogSite cmdlet. The properties of the site object are returned by the Get-LogSite cmdlet.

High Level Operations

A high level operation object represents a logged configuration change performed from Desktop Studio, Desktop Director or a PowerShell Script.

The XenDesktop consoles log high level operations when:

1) Executing operations which performs configuration changes.
2) Executing operations which performs administration related activities which may affect site configuration.
PowerShell scripts which carry out customized configuration changes can also log high level operations via cmdlets Start-LogHighLevelOperation and Stop-LogHighLevelOperation.

Low Level Operations

A low level operation object represents a logged configuration change performed by a service. One or more low level operation objects are used to record the actions performed by a services in order to fulfil a high level operation initiated from the consoles, or from PowerShell scripts.

Low level operations in the system are returned by cmdlet Get-LogLowLevelOperation.

Operation Details

A low level operation performed by a service can affect a number of individual objects, or a number of properties on an object. An operation detail log records each individual change to an object. This includes the creation and deletion of the object, as well as changes to individual properties of the object.

One or more operation detail objects are used to record specific changes to each object that is affected by a low level service operation.

Operation details are included in the data returned from the Get-LogLowLevelOperation cmdlet.

High Level Operations, Low Level Operations and Operation Details are arranged in a hierarchy. A High Level Operation can have multiple Low Level Operations, and each Low Level Operation can have multiple Operation Details.

Setting up a separate logging database -------------------------------------- After creating the database on the database server, the logging database can be setup and configured for use by:

1) Generating the database schema, and applying it the logging database
2) Configuring the configuration logging service to use the new logging database.

The logging database schema can be generated from the Get-LogDBSchem cmdlet, as illustrated below:
Get-LogDBSchema -DatabaseName "loggingDB"
  -ServiceGroupName "service group name"
  -ScriptType Database
  -LocalDatabase:$LocalDB
  -DataStore Logging

The configuration logging service can be configured to use the logging database with the Set-LogDBConnection cmdlet, as illustrated below:

    Set-LogDBConnection -DataStore Logging -DBConnection $null
    Set-LogDBConnection -DataStore Logging -DBConnection "new logging db connection string"

Configuration Logging site settings ----------------------------------------------

On the Site object:

  o The 'State' setting allows configuration logging to be disabled, enabled, or made mandatory.

  o The 'Locale' setting specifies the language in which configuration logging data text will be stored.

See Get-LogSite cmdlet help for further information on these settings.

This locale setting applies to the text description that is associated with each log, e.g. Create Catalog. It doesn't apply to other textual information in the log like the names of parameters passed to operations, e.g. CatalogName.

This localisation is applied when the data is logged, and not when the logs are viewed later. For example, logs which are created in English will be displayed in English on an end user system which may be configured with a different locale.
XenDesktop - Advanced Dataset Filtering

Describes the common filtering options for XenDesktop cmdlets.

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

```powershell
Get-<Noun> | Where { $_.Size = 'Small' } | Sort 'Date' | Select -First 10
```

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

```powershell
Get-<Noun> -Size 'Small' -SortBy 'Date' -MaxRecordCount 10
```

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

- **MaxRecordCount <int>**

  Specifies the maximum number of results to return.

  For example, to return only the first nine results use:

  ```powershell
  Get-<Noun> -MaxRecordCount 9
  ```

  If not specified, only the first 250 records are returned, and if more are available, a warning is produced:
WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

To retrieve all records, specify a large number for -MaxRecordCount. As the value is an integer, you can use the following:

```
Get-<Noun> -MaxRecordCount [int]:MaxValue
```

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
```

... 

```
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<< -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (;) [Get-<Noun>], PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

```
$count = $error[0].TotalAvailableResultCount
```

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.
Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values. Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

-SortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter 'Name -like "High*" -or (Priority -eq 1 -and Severity -ge 2)'

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

EXAMPLES

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company "citrix" -Product '[X]EN*"
Get-<Noun> -Product "Xen*" -Company "CITRIX"
Get-<Noun> -Filter { Company -eq 'Citrix' -and Product -like 'Xen*' }
See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```powershell
Get-<Noun> -Company "Abc[*]"                 # Matches Abc*
Get-<Noun> -Company "Abc`*"                  # Matches Abc*
Get-<Noun> -Filter { Company -eq "Abc`*" }   # Matches Abc*
Get-<Noun> -Filter { Company -eq "A`B`C`" }  # Matches A`B`C
```

Simple filtering by numbers, booleans, and TimeSpanS perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```powershell
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

```powershell
Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```powershell
Get-<Noun> -Filter 'Enabled'      # Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled' # Equivalent to 'Enabled -eq $false'
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```powershell
Get-<Noun> -Shape [Shapes]:Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle" }
Get-<Noun> -Filter { Shape -like 'C' } 

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z" }
$d = [DateTime]"2010-08-23T12:30:00.0Z"
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge '-2' } # Two days ago
Get-<Noun> -Filter { StartTime -ge '-1:30' } # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge '-0:0:30' } # 30 seconds ago

ARRAY PROPERTIES

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings. Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*" }
Get-<Noun> -Filter { Users -contains "Fred*" }

You can also use the singular form with -Filter to search using other operators:

# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter { User -eq "Frederick" }
# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter { User -lt 'F' }

COMPLEX EXPRESSIONS
When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

```powershell
Get-<Noun> -Filter { Shape -eq 'Circle' -or Shape -eq 'Square' }
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }
```

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

```powershell
Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue' -and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue' -and Shape -eq 'Circle') }
```

### PAGING

The simplest way to page through data is to use the -Skip and -MaxRecordCount parameters. So, to read the first three pages of data with 10 records per page, use:

```powershell
Get-<Noun> -Skip 0  -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 10 -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 20 -MaxRecordCount 10 <other filtering criteria>
```

You must include the same filtering criteria on each call, and ensure that the data is sorted consistently.

The above approach is often acceptable, but as each call performs an independent query, data changes can result in records being skipped or appearing twice. One approach to improve this is to sort by a unique id field and then start the search for the next page at the unique id after the last unique id of the previous page. For example:

```powershell
# Get the first page
Get-<Noun> -MaxRecordCount 10 -SortBy SerialNumber

SerialNumber    ...
-------------   ----
A120004
A120007
... 7 other records ...
A120900

# Get the next page
Get-<Noun> -MaxRecordCount 10 -Filter { FirstName -gt 'A120900' }

SerialNumber    ...
-------------   ----
```
FILTER SYNTAX DEFINITION

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= 

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

  <Component>

<Component> ::= <NotOperator> <Factor> |

  <Factor>

<Factor> ::= "(" <ComponentList> ")" |

  <PropertyName> <ComparisonOperator> <Value> |
  <PropertyName>

<AndOrOperator> ::= "-and" | "-or"

<NotOperator> ::= "-not" | "!

<ComparisonOperator> ::= "-eq" | "-ne" | "-le" | "-ge" | "-lt" | "-gt" |
  "-like" | "-notlike" | "-contains" | "-notcontains" |
  "-in" | "-notin"

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |
  <scalar variable> | <array variable> | "$null" | "$true" | "$false"
Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30" means 1 hour and 30 minutes ago.
about_logical_operators

Oct 10, 2013

TOPIC

about_Logical_Operators

SHORT DESCRIPTION

Describes the operators that connect statements in Windows PowerShell.

LONG DESCRIPTION

The Windows PowerShell logical operators connect expressions and statements, allowing you to use a single expression to test for multiple conditions.

For example, the following statement uses the and operator and the or operator to connect three conditional statements. The statement is true only when the value of $a is greater than the value of $b, and either $a or $b is less than 20.

```
($a -gt $b) -and (($a -lt 20) -or ($b -lt 20))
```

Windows PowerShell supports the following logical operators.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>-and</td>
<td>Logical and. TRUE only when both statements are TRUE.</td>
<td>(1 -eq 1) -and (1 -eq 2)</td>
</tr>
<tr>
<td>-or</td>
<td>Logical or. TRUE when either or both statements are TRUE.</td>
<td>(1 -eq 1) -or (1 -eq 2)</td>
</tr>
<tr>
<td>-xor</td>
<td>Logical exclusive or. TRUE only when one of the statements is TRUE and the other is FALSE.</td>
<td>(1 -eq 1) -xor (2 -eq 2)</td>
</tr>
<tr>
<td>-not</td>
<td>Logical not. Negates the statement that follows it.</td>
<td>-not (1 -eq 1)</td>
</tr>
<tr>
<td>!</td>
<td>Logical not. Negates the statement that follows it.</td>
<td>!(1 -eq 1)</td>
</tr>
</tbody>
</table>

(Same as -not)
Note: The previous examples also use the equal to comparison operator (-eq). For more information, see about_Comparison_Operators. The examples also use the Boolean values of integers. The integer 0 has a value of FALSE. All other integers have a value of TRUE.

The syntax of the logical operators is as follows:

```plaintext
<statement> {-AND | -OR | -XOR} <statement>
{! | -NOT} <statement>
```

Statements that use the logical operators return Boolean (TRUE or FALSE) values.

The Windows PowerShell logical operators evaluate only the statements required to determine the truth value of the statement. If the left operand in a statement that contains the and operator is FALSE, the right operand is not evaluated. If the left operand in a statement that contains the or statement is TRUE, the right operand is not evaluated. As a result, you can use these statements in the same way that you would use the If statement.

SEE ALSO

about_Operators

Compare-Object

about_Comparison_operators

about_If
Export-LogReportCsv

Oct 10, 2013
Exports Configuration Logging data into a CSV file.

Syntax

Export-LogReportCsv -OutputFile <String> [-StartDateRange <DateTime>] [-EndDateRange <DateTime>] [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

This cmdlet exports the Configuration Logging data into a CSV data file. The hierarchical logging data is flattened into a single CSV 'table'. The content of CSV file is not intended to be human-readable. It is meant to be input data for external reporting or manipulation tools (for example, a spreadsheet application).

Related topics

Export-LogReportHtml

Parameters

-OutputFile <String>

Specifies the path to a file where the CSV data will be saved.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | false |

-StartDateRange <DateTime>

Specifies the start time of the earliest operation to include.

| Required? | false |
| Default Value | DateTime.Min |
| Accept Pipeline Input? | false |

-EndDateRange <DateTime>

Specifies the end time of the latest operation to include.

| Required? | false |
| Default Value | DateTime.UtcNow |
| Accept Pipeline Input? | false |

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? |
Accept Pipeline Input? false

Examples

---------------------------- EXAMPLE 1 ----------------------------
C:\PS> Export-LogReportCsv -OutputFile "c:\MyReports\LoggingData.csv"
Export all logged operations to a csv file.

---------------------------- EXAMPLE 2 ----------------------------
C:\PS> Export-LogReportCsv -OutputFile "c:\MyReports\LoggingData.csv" -StartDateRange "2012-12-21 09:00"
Export to a CSV file logged operations started on or after a specified datetime.

---------------------------- EXAMPLE 3 ----------------------------
C:\PS> Export-LogReportCsv -OutputFile "c:\MyReports\LoggingData.csv" -StartDateRange "2012-12-21 09:00" -EndDateRange "2012-12-31 18:00"
Export to a CSV file logged operations started and completed between a date range.
Export-LogReportHtml

Oct 10, 2013
Exports Configuration Logging data into a HTML report.

Syntax
Export-LogReportHtml -OutputDirectory <String> [-StartDateRange <DateTime>] [-EndDateRange <DateTime>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
This cmdlet exports the Configuration Logging data into a HTML report. The report consists of two HTML files:
- Summary.Html - this shows summary information from the high level operation logs.
- Details.Html - this shows additional logging data from the low level operation and operation detail logs.

Hyperlinks in summary.html allow drill-down into the associated low level logging data contained within details.html.

Related topics
Export-LogReportCsv

Parameters
- **OutputDirectory**<String>
  Specifies the path to a directory where the HTML report files will be saved.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>OutputDirectory</td>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **StartDateRange**<DateTime>
  Specifies the start time of the earliest operation to include.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>StartDateRange</td>
<td>false</td>
<td>DateTime.Min</td>
<td>false</td>
</tr>
</tbody>
</table>

- **EndDateRange**<DateTime>
  Specifies the end time of the latest operation to include.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>EndDateRange</td>
<td>false</td>
<td>DateTime.UtcNow</td>
<td>false</td>
</tr>
</tbody>
</table>
**AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Export-LogReportHtml -OutputDirectory "c:\MyReports"
Export all logged operations to HTML.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Export-LogReportHtml -OutputDirectory "c:\MyReports" -StartDateRange "2012-12-21 09:00"
Export to a HTML logged operations started on or after a specified datetime..

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Export-LogReportHtml -OutputDirectory "c:\MyReports" -StartDateRange "2012-12-21 09:00" -EndDateRange "2012-12-31 18:00"
Export to HTML logged operations started and completed between a date range.
Get-LogDataStore

Oct 10, 2013

Gets details for each of the ConfigurationLogging data stores.

Syntax

Get-LogDataStore [-AdminAddress <String>] [&<CommonParameters>]

Detailed Description

Returns an object for each of the ConfigurationLogging data stores describing the connection string, data store name, db type, provider, schema name, and DB status.

A database connection must be configured in order for this command to be used if the service has a secondary data store. This is not required for the site data store.

Related topics

Reset-LogDataStore

Parameters

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.ConfigurationLogging.Sdk.DataStoreConfiguration

An object describing the connection string, data store name, database type, provider, schema name and database status.

Notes

If the command fails, the following errors can be returned.

Error Codes

---------

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-LogDataStore

ConnectionString : Server=\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True
DataStore : Site
DatabaseType : SqlServer
Provider : MSSQL
SchemaName : LogSiteSchema
Status : OK

ConnectionString : 
DataStore : Secondary 
DatabaseType : SqlServer 
Provider : MSSQL 
SchemaName : LogSecondarySchema 
Status : DBUnconfigured

Get the database connection string for the ConfigurationLogging Service.
Get-LogDBConnection

Oct 10, 2013
Gets the database string for the specified data store used by the ConfigurationLogging Service.

Syntax

Get-LogDBConnection [-DataStore <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns the database connection string for the specified data store.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

Related topics

Get-LogServiceStatus
Get-LogDataStore
Set-LogDBConnection
Test-LogDBConnection

Parameters

-DataStore

Specifies the logical name of the data store for the ConfigurationLogging Service. Can be either be 'Site' or the logical name of the secondary data store.

| Required? | false |
| Default Value | Site |
| Accept Pipeline Input? | false |

-AdminAddress

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values
The database connection string configured for the ConfigurationLogging Service.

Notes

If the command fails, the following errors can be returned.

Error Codes

------------

NoDBConnections

The database connection string for the ConfigurationLogging Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-LogDBConnection

Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True

Get the database connection string for the ConfigurationLogging Service.
Get-LogDBSchema

Oct 10, 2013
Gets a script that creates the ConfigurationLogging Service database schema for the specified data store.

Syntax


Detailed Description

Gets SQL scripts that can be used to create a new ConfigurationLogging Service database schema, add a new ConfigurationLogging Service to an existing site, remove a ConfigurationLogging Service from a site, or create a database server logon for a ConfigurationLogging Service. If no Sid parameter is provided, the scripts obtained relate to the currently selected ConfigurationLogging Service instance, otherwise the scripts relate to ConfigurationLogging Service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied. The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to a ConfigurationLogging SDK cmdlet. The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured. The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SMDCMD mode'. The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to ConfigurationLogging Service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to ConfigurationLogging Service roles

If ScriptType is Evict, the returned script contains:

- Removal of ConfigurationLogging Service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

If the service uses two data stores they can exist in the same database. You do not need to configure a database before using this command.

Related topics
Get-LogDataStore

Set-LogDBConnection

Parameters

- **DatabaseName**<String>

Specifies the name of the database for which the schema will be generated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ServiceGroupName**<String>

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the ConfigurationLogging services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ScriptType**<ScriptTypes>

Specifies the type of database script returned. Available script types are:

**Database**

Returns a full database script that can be used to create a database schema for the ConfigurationLogging Service in a database instance that does not already contain a schema for this service. The DatabaseName and ServiceGroupName parameters must be specified to create a script of this type.

**Instance**

Returns a permissions script that can be used to add further ConfigurationLogging services to an existing database instance that already contains the full ConfigurationLogging service schema, associating the services to the Service Group. The Sid parameter can optionally be specified to create a script of this type.

**Login**

Returns a database logon script that can be used to add the required logon accounts to an existing database instance that contains the ConfigurationLogging Service schema. This is used primarily when creating a mirrored database environment. The DatabaseName parameter must be specified to create a script of this type.

**Evict**

Returns a script that can be used to remove the specified ConfigurationLogging Service from the database entirely. The DatabaseName and Sid parameters must be specified to create a script of this type.
Required? | false  
---|---
Default Value | Database  
Accept Pipeline Input? | false  

-**LocalDatabase**<SwitchParameter>

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for Configuration Logging services.

Required? | false  
---|---
Default Value |  
Accept Pipeline Input? | false  

-**Sid**<String>

Specifies the SID of the controller on which the Configuration Logging Service instance to remove from the database is running.

Required? | false  
---|---
Default Value |  
Accept Pipeline Input? | true (ByValue)  

-**DataStore**<String>

Specifies the logical name of the data store for the Configuration Logging Service. Can be either be 'Site' or the logical name of the secondary data store.

Required? | false  
---|---
Default Value | Site  
Accept Pipeline Input? | false  

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

Required? | false  
---|---
Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default.  
Accept Pipeline Input? | false  

Return Values
A string containing the required SQL script for application to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

Create the database schema.

Create the user and the role (providing the schema does not already exist).

Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

Create the user and the role (providing the schema does not already exist).

Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned.

Error Codes

GetSchemasFailed
The database schema could not be found.

ActiveDirectoryAccountResolutionFailed
The specified Active Directory account or Group could not be found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-LogDBSchema -DatabaseName MyDB -ServiceGroupName MyServiceGroup > c:\LogSchema.sql
Get the full database schema for site data store of the ConfigurationLogging Service and copy it to a file called 'c:\LogSchema.sql'.

This script can then be used to create the schema in a pre-existing database named 'MyDB' that does not already contain a ConfigurationLogging Service site schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-LogDBSchema -DatabaseName MyDB -scriptType Login > c:\ConfigurationLoggingLogins.sql
Get the logon scripts for the ConfigurationLogging Service.

-------------------------- EXAMPLE 3 --------------------------

c:\PS>Get-LogDBSchema -DatabaseName MyDB -ServiceGroupName MyServiceGroup -DataStore Secondary > c:\LogSchema.sql
Get the full database schema for the secondary data store of the ConfigurationLogging Service and copy it to a file called 'c:\LogSecondarySchema.sql'.

This script can then be used to create the schema in a pre-existing database named 'MyDB' that does not already contain a ConfigurationLogging Service secondary schema.
Get-LogDBVersionChangeScript

Oct 10, 2013

Gets a script that updates the ConfigurationLogging Service database schema.

Syntax

Get-LogDBVersionChangeScript -DatabaseName <String> -TargetVersion <Version> [-DataStore <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns a database script that can be used to upgrade or downgrade the site or secondary schema for the ConfigurationLogging Service from the current schema version to a different version.

Related topics

Get-LogInstalledDBVersion

Parameters

-DatabaseName <String>

Specifies the name of the database instance to which the update applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-TargetVersion <Version>

Specifies the version of the database you want to update to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-DataStore <String>

Specifies the logical name of the data store for the ConfigurationLogging Service. Can be either be 'Site' or the logical name of the secondary data store.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Site</td>
<td>false</td>
</tr>
</tbody>
</table>
### -AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**

System.Management.Automation.PSObject

A PSObject containing the required SQL script for application to a database.

**Notes**

The PSObject returned by this cmdlet contains the following properties:

-- Script The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

-- NeedExclusiveAccess Indicates whether all services in the service group must be shut down during the update or not.

-- CanUndo Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any ConfigurationLogging services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-LogServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned.

**Error Codes**

---------

NoOp

The operation was successful but had no effect.

NoDBConnections
The database connection string for the ConfigurationLogging Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $update = Get-LogDBVersionChangeScript -DatabaseName MyDb -TargetVersion 1.0.75.0
C:\PS> $update.Script > update_75.sql

Gets an SQL update script to update the current schema to version 1.0.75.0. The resulting update_75.sql script is suitable for direct use with the SQL Server SQLCMD utility.
Get-LogHighLevelOperation

Oct 10, 2013
Gets high level operations

Syntax


Detailed Description

Retrieves high level operations matching the specified criteria. If no parameters are specified this cmdlet returns all high level operations.

Related topics

Start-LogHighLevelOperation
Stop-LogHighLevelOperation
Get-LogLowLevelOperation

Parameters

-Id <Guid>

Gets the high level operation with the specified identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Text <String>

Gets high level operations with the specified text

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-StartTime <DateTime>

Gets high level operations with the specified start time


<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Source <String>**

Gets high level operations with the specified source.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-EndTime <DateTime>**

Gets high level operations with the specified end time.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-IsSuccessful <Boolean>**

Gets high level operations with the specified success indicator.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-User <String>**

Gets high level operations logged by the specified user.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminMachineIP <String>**

Gets high level operations logged from the machine with the specified IP address.
### -OperationType

**Description:** Gets high level operations with the specified operation type. Values can be:

- AdminActivity - to get operations which log administration activity.
- ConfigurationChange - to get operations which log configuration changes.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -TargetType

**Description:** Gets high level operations with the specified target type. The target type describes the type of object that was the target of the configuration change. For example, "Session" or "Machine".

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -ReturnTotalRecordCount

**Description:** When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Log_Filtering for details.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

### -MaxRecordCount

**Description:** Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |
Accept Pipeline Input? | false

-Skip<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |

Accept Pipeline Input? | false

-SortBy<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |

Accept Pipeline Input? | false

-Filter<String>

Gets records that match a PowerShell-style filter expression. See about_Log_Filtering for details.

| Required? | false |
| Accept Pipeline Input? | false |

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |

Accept Pipeline Input? | false

Return Values

Citrix.ConfigurationLogging.Sdk.HighLevelOperation
The returned HighLevelOperation object has the following properties:

- **Id** - The unique identifier of the operation.
- **Text** - A description of the operation.
- **StartTime** - The date and time that the operation started.
- **EndTime** - The date and time that the operation completed. This will be null if the operation is still in progress, or if the operation never completed.
- **IsSuccessful** - Indicates whether the operation completed successfully or not. This will be null if the operation is still in progress, or if the operation never completed.
- **User** - The identifier of the administrator who performed the operation.
- **AdminMachineIP** - The IP address of the machine that the operation was initiated from.
- **Source** - Identifies the XenDesktop console, or custom script, where the operation was initiated from. For example, "Desktop Studio", "Desktop Director", "My custom script".
- **OperationType** - The operation type. This can be 'AdminActivity' or 'ConfigurationChange'.
- **TargetTypes** - Identifies the type of objects that were affected by the operation. For example, "Catalog" or "Desktop","Machine".
- **Parameters** - The names and values of the parameters that were supplied for the operation.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-LogHighLevelOperation
Get all high level operations

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-LogHighLevelOperation -OperationType ConfigurationChange
Get high level operations which log configuration changes.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Get-LogHighLevelOperation -OperationType AdminActivity
Get high level operations which log administration activities.

-------------------------- EXAMPLE 4 --------------------------

C:\PS> Get-LogHighLevelOperation -Filter{ StartTime -ge "2013-02-27 09:00:00" -and EndTime -le "2013-02-27 18:00:00"}
Use advanced filtering to get high level operations with a start time on or after "2013-02-27 09:00:00" and an end time on or before "2013-02-27 18:00:00".

-------------------------- EXAMPLE 5 --------------------------

C:\PS> Get-LogHighLevelOperation -EndTime $null
C:\PS> Get-LogHighLevelOperation -IsSuccessful $null
Either of these commands will get high level operations which have not yet been completed.

-------------------------- EXAMPLE 6 --------------------------

C:\PS> Get-LogHighLevelOperation -User "DOMAIN\UserName"
Get high level operations performed by user "DOMAIN\UserName".
Get-LogInstalledDBVersion

Oct 10, 2013
Gets a list of all available database schema versions for the Configuration Logging Service.

Syntax


Detailed Description

Returns the current version of the Configuration Logging Service database schema, if no flags are set, otherwise returns versions for which upgrade or downgrade scripts are available and have been stored in the database.

Related topics

Parameters

- **Upgrade**<SwitchParameter>

  Specifies that only schema versions to which the current database version can be updated should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Downgrade**<SwitchParameter>

  Specifies that only schema versions to which the current database version can be reverted should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **DataStore**<String>

  Specifies the database connection logical name the schema script should be returned for. The parameter is optional.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.Version

The Get-LoginstalledDbVersion command returns objects containing the new definition of the ConfigurationLogging Service database schema version.

Major <Integer>

Minor <Integer>

Build <Integer>

Revision <Integer>

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the ConfigurationLogging Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.
PermissionDenied
You do not have permission to execute this command.
AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.
CommunicationError
There was a problem communicating with the remote service.
ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-LogInstalledDBVersion

<table>
<thead>
<tr>
<th>Major</th>
<th>Minor</th>
<th>Build</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Get the currently installed version of the ConfigurationLogging Service database schema.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Get-LogInstalledDBVersion -Upgrade

<table>
<thead>
<tr>
<th>Major</th>
<th>Minor</th>
<th>Build</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Get the versions of the ConfigurationLogging Service database schema for which upgrade scripts are supplied.
Get-LogLowLevelOperation

Oct 10, 2013
Gets low level operations

Syntax


Detailed Description

Retrieves low level operations matching the specified criteria. If no parameters are specified this cmdlet returns all low level operations.

Related topics

Get-LogLowLevelOperation

Parameters

-Id <Guid>

Gets the low level operation with the specified identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-HighLevelOperationId <Guid>

Gets low level operations for the high level operation with the specified identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-StartTime <DateTime>

Gets low level operations with the specified start time.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input?  false

**-EndTime**<DateTime>

Gets low level operations with the specified end time.

Required?  false
Default Value
Accept Pipeline Input?  false

**-IsSuccessful**<Boolean>

Gets low level operations with the specified success indicator.

Required?  false
Default Value
Accept Pipeline Input?  false

**-User**<String>

Gets low level operations logged by the specified administrator.

Required?  false
Default Value
Accept Pipeline Input?  false

**-AdminMachineIP**<String>

Gets low level operations logged from the machine with the specified IP address.

Required?  false
Default Value
Accept Pipeline Input?  false

**-Text**<String>

Gets low level operations with the specified text.

Required?  false
Default Value
Accept Pipeline Input? | false
---|---
**-Source**<String>
Gets low level operations with the specified source.

Required? | false
Default Value |
Accept Pipeline Input? | false

**-SourceSdk**<String>
Gets low level operations logged from the SDK with the specified identifier.

Required? | false
Default Value |
Accept Pipeline Input? | false

**-OperationType**<OperationType>
Gets low level operations with the specified operation type. Values can be:
- AdminActivity - to get operations which log administration activity.
- ConfigurationChange - to get operations which log configuration changes.

Required? | false
Default Value |
Accept Pipeline Input? | false

**-TargetType**<String>
Gets low level operations with the specified target type. The target type describes the type of object that was the target of the configuration change. For example, "Session" or "Machine".

Required? | false
Default Value |
Accept Pipeline Input? | false

**-ReturnTotalRecordCount**<SwitchParameter>
When specified, the cmdlet outputs an error record containing the number of records available. This error record is...
additional information and does not affect the objects written to the output pipeline. See about_Log_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-MaxRecordCount (Int32)**

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Skip (Int32)**

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SortBy (String)**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter (String)**

Gets records that match a PowerShell-style filter expression. See about_Log_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**: String

  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.ConfigurationLogging.Sdk.LowLevelOperation

The returned LowLevelOperation object has the following properties:

- **Id** - The unique identifier of the operation.
- **Text** - A description of the operation.
- **HighLevelOperationId** - The unique identifier of the related high level operation.
- **StartTime** - The date and time that the operation started.
- **EndTime** - The date and time that the operation completed. This will be null if the operation is still in progress, or if the operation never completed.
- **IsSuccessful** - Indicates whether the operation completed successfully or not. This will be null if the operation is still in progress, or if the operation never completed.
- **AdminSid** - The identifier of the administrator who performed the operation.
- **AdminMachineIP** - The IP address of the machine that the operation was performed on.
- **Source** - The name of the XenDesktop service that the operation was performed on; for example, "MachineCreation", "DelegatedAdmin".
- **SourceSdk** - The identifier of the XenDesktop service SDK through which the operation was performed; for example, "Prov", "Admin".
- **OperationType** - The operation type. This can be 'AdminActivity' or 'ConfigurationChange'.
- **TargetTypes** - Identifies the type of objects that were affected by the operation. For example, "Catalog" or "Desktop","Machine".
- **Parameters** - The names and values of the parameters that were supplied for the operation.
- **Details** - A collection of OperationDetail objects containing specific information about each object affected by the operation.

Each OperationDetail object in the 'Details' collection has the following properties:
- **TargetUid** - The unique identifier of the target object affected by the operation.
- **TargetName** - The name of the target object affected by the operation.
- **TargetType** - The type of the target object.
- **Text** - The description of the operation performed on the target object.
- **StartTime** - The date and time that the operation started.
- **EndTime** - The date and time that the operation completed. This will be null if the operation is still in progress, or if the operation never completed.
- **IsSuccessful** - Indicates whether the operation completed successfully or not. This will be null if the operation is still in progress, or if the operation didn't complete.

The following properties will be set if the operation changed a property on the object:

- **PropertyName** - The name of the changed property.
- **NewValue** - The new property value.
- **PreviousValue** - The previous property value.
- **AddValue** - If the object property contains a set of values, this specifies the new value which was added to the set.
- **RemoveValue** - If the object property contains a set of values, this specifies the value which was removed from the set.

Examples

------------- EXAMPLE 1 -------------
C:\PS> Get-LogLowLevelOperation
Get all low level operations

------------- EXAMPLE 2 -------------
C:\PS> Get-LogLowLevelOperation -OperationType ConfigurationChange
Get low level operations which log configuration changes.

------------- EXAMPLE 3 -------------
C:\PS> Get-LogLowLevelOperation -OperationType AdminActivity
Get low level operations which log administration activities.

------------- EXAMPLE 4 -------------
C:\PS> Get-LogLowLevelOperation -Filter{ StartTime -ge "2013-02-27 09:00:00" -and EndTime -le "2013-02-27 18:00:00"}
Use advanced filtering to get low level operations with a start time on or after "2013-02-27 09:00:00" and an end time on or before "2013-02-27 18:00:00".

------------- EXAMPLE 5 -------------
C:\PS> Get-LogLowLevelOperation -EndTime $null
C:\PS> Get-LogLowLevelOperation -IsSuccessful $null
Either of these commands will get low level operations which have not yet been completed.

-------------------------- EXAMPLE 6 --------------------------

C:\PS> Get-LogLowLevelOperation -User "DOMAIN\UserName"
Get low level operations performed by user "DOMAIN\UserName".
Get-LogService

Oct 10, 2013
Gets the service record entries for the ConfigurationLogging Service.

Syntax


Detailed Description

Returns instances of the ConfigurationLogging Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Related topics

Parameters

- **-Property <String[]>**

  Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-ReturnTotalRecordCount <SwitchParameter>**

  When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Log_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>False</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-MaxRecordCount <Int32>**

  Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>250</td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | false
---|---

**-Skip[Int32]**

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | false

**-SortBy[String]**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | false

**-Filter[String]**

Gets records that match a PowerShell-style filter expression. See about_Log_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>

Accept Pipeline Input? | false

**-AdminAddress[String]**

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | false

**Return Values**

Citrix.ConfigurationLogging.Sdk.Service
The Get-LogServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.
Notes

If the command fails, the following errors can be returned.

Error Codes
------------

UnknownObject
One of the specified objects was not found.

PartialData
Only a subset of the available data was returned.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase
The query required to get the database was not defined.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS> Get-LogService

Uid  : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
SID                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {MySiteService1, MySiteService2...}

Get all the instances of the ConfigurationLogging Service running in the current service group.
Get-LogServiceAddedCapability

Oct 10, 2013
Gets any added capabilities for the ConfigurationLogging Service on the controller.

Syntax
Get-LogServiceAddedCapability [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Enables updates to the ConfigurationLogging Service on the controller to be detected.
You do not need to configure a database connection before using this command.

Related topics
Parameters

- **-AdminAddress**<String>
  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
System.String
String containing added capabilities.

Notes
If the command fails, the following errors can be returned.

Error Codes

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-LogServiceAddedCapability
Get the added capabilities of the ConfigurationLogging Service.
Get-LogServiceInstance

Oct 10, 2013
Gets the service instance entries for the ConfigurationLogging Service.

Syntax

Get-LogServiceInstance [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns service interfaces published by the instance of the ConfigurationLogging Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Related topics

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

Return Values

Citrix.ConfigurationLogging.Sdk.ServiceInstance

The Get-LogServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.
ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Log.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.ConfigurationLogging.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

---------

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-LogServiceInstance

Address            : http://MyServer.com:80/Citrix/ConfigurationLoggingService
Binding            : wcf_HTTP_kerb
InterfaceType      : SDK
Metadata           :
MetadataMap        :
ServiceAccount     : ENG\MyAccount$
ServiceAccountSid  : S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName   : MyServiceGroup
ServiceGroupUid    : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceUid : 00000000-0000-0000-0000-000000000000
ServiceType        : Log
Version            : 1

Address            : http://MyServer.com:80/Citrix/ConfigurationLoggingService/IServiceApi
Binding            : wcf_HTTP_kerb
InterfaceType      : InterService
Metadata           :
MetadataMap        :
ServiceAccount     : ENG\MyAccount
ServiceAccountSid  : S-1-5-21-2406005612-3133289213-1653143164

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
ServiceGroupName : MyServiceGroup
ServiceGroupUid : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceUid : 00000000-0000-0000-0000-000000000000
ServiceType : Log
Version : 1

Get all instances of the ConfigurationLogging Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.
Get-LogServiceStatus

Oct 10, 2013
Gets the current status of the ConfigurationLogging Service on the controller.

Syntax

Get-LogServiceStatus [-AdminAddress <String>] [CommonParameters]

Detailed Description

Enables the status of the ConfigurationLogging Service on the controller to be monitored. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured.

Related topics

Get-LogDataStore
Set-LogDBConnection
Test-LogDBConnection
Get-LogDBConnection
Get-LogDBSchema

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values


The Get-LogServiceStatus command returns an object containing the status of the ConfigurationLogging Service together with extra diagnostics information.

DBUnconfigured

The ConfigurationLogging Service does not have a database connection configured.

DBRejectedConnection
The database rejected the logon attempt from the ConfigurationLogging Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the ConfigurationLogging Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBNewerVersionThanService

The version of the ConfigurationLogging Service currently in use is incompatible with the version of the ConfigurationLogging Service schema on the database. Upgrade the ConfigurationLogging Service to a more recent version.

DBOlderVersionThanService

The version of the ConfigurationLogging Service schema on the database is incompatible with the version of the ConfigurationLogging Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The ConfigurationLogging Service is running and is connected to a database containing a valid schema.

Failed

The ConfigurationLogging Service has failed.

Unknown

(0) The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-LogServiceStatus

DBUnconfigured
Get the current status of the ConfigurationLogging Service.
Get-LogSite

Oct 10, 2013
Gets global configuration logging settings.

Syntax

Get-LogSite [-AdminAddress <String>] [<-CommonParameters>]

Detailed Description

This cmdlet retrieves the global configuration logging settings.

Related topics

Set-LogSite

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.ConfigurationLogging.Sdk.Site

Get-LogSite returns an object containing the following properties:

- Locale - the current language that logs should be recorded in. Can be: 'en', 'ja', 'zh-CN', 'de', 'es' or 'Fr'.
- State - the current state of configuration logging. Can be: Enabled, Disabled, Mandatory or NotSupported.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-LogSite
Gets configuration logging site settings.
Get-LogSummary

Oct 10, 2013
Gets operations logged within time intervals inside a date range.

Syntax

Get-LogSummary [-StartDateRange <DateTime>] [-EndDateRange <DateTime>] [-IntervalSeconds <Int64>] [-GetLowLevelOperations] [-IncludeIncomplete] [-OperationType <OperationType>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Get-LogSummary cmdlet retrieves summary counts of operations logged within time intervals inside a date range. The returned data indicates the rate at which configuration changes and activities were performed out within a time period.

Related topics

Get-LogHighLevelOperation

Get-LogLowLevelOperation

Parameters

- **-StartDateRange**<DateTime>

  Specifies the start of the summary period date range

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>1900-01-01 00:00:00</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-EndDateRange**<DateTime>

  Specifies the end of the summary period date range.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>DateTime.UtcNow</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-IntervalSeconds**<Int64>

  Specifies the size, in seconds, of each time interval required within the summary date range. If this is not specified, is null, zero or exceeds the specified date range, it defaults to the total number of seconds between EndDateRange and StartDateRange.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Total number of seconds in the EndDateRange and StartDateRange time span.</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-GetLowLevelOperations**<SwitchParameter>
Specifies if the cmdlet should return low level operation summary counts.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>$false - high level operations counts are returned.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **IncludeIncomplete**<SwitchParameter>

Specifies if incomplete operations should be included in the returned summary counts.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>$false - incomplete operations are excluded.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **OperationType**<OperationType>

Specifies the type of logged operations to include. Values can be 'AdminActivity' or 'ConfigurationChange'

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


The summary data is returned as a collection of dictionary items. The 'Key' value of each dictionary item specifies the start of the time interval within the overall summary date range. The 'Value' data in each dictionary item contains the count of operations which were started within that time interval.

Notes

If the specified summary date range and interval period will result in more than 50,000 intervals being returned Get-LogSummary will generate an error and abort the operation.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> $logSummary = Get-LogSummary
Get a summary of all completed high level operations. The returned log summary collection will contain a single item for the time period spanning the entire [1900-01-01 00:00:00]-[UtcNow] date range. e.g.

Key ==> Value
01/01/1900 00:00:00 ==> 41

-------------------------- EXAMPLE 2 --------------------------
C:\PS> $daily = 60*60*24
C:\PS> [DateTime]$startRange = "2013-02-01 14:50:39"
C:\PS> [DateTime]$endRange = $startRange.AddDays(14)
C:\PS> Get-LogSummary -StartDateRange $startRange -EndDateRange $endRange -intervalSeconds $daily
Gets a summarised count of completed high level operations logged over two weeks, at daily intervals. The returned log summary collection will contain multiple items; one for each day in the summary date range. e.g.

Key ==> Value
01/02/2013 14:50:39 ==> 0
02/02/2013 14:50:39 ==> 4
03/02/2013 14:50:39 ==> 21
04/02/2013 14:50:39 ==> 0
05/02/2013 14:50:39 ==> 0
06/02/2013 14:50:39 ==> 0
07/02/2013 14:50:39 ==> 5
08/02/2013 14:50:39 ==> 0
09/02/2013 14:50:39 ==> 0
10/02/2013 14:50:39 ==> 0
11/02/2013 14:50:39 ==> 0
12/02/2013 14:50:39 ==> 7
13/02/2013 14:50:39 ==> 0
14/02/2013 14:50:39 ==> 0
15/02/2013 14:50:39 ==> 12

-------------------------- EXAMPLE 3 --------------------------
C:\PS> $hourly = 60*60
C:\PS> [DateTime]$startRange = "2013-02-03 00:00:00"
C:\PS> [DateTime]$endRange = "2013-02-03 23:59:59"
C:\PS> Get-LogSummary -StartDateRange $startRange -EndDateRange $endRange -intervalSeconds $hourly -GetLowLevelOperations
Gets a summarised count of completed low level operations logged during a day, at hourly intervals. The returned log summary collection will contain multiple items; one for each hour in the summary date range. e.g.

Key ==> Value
04/03/2013 00:00:00 ==> 12
04/03/2013 01:00:00 ==> 10
04/03/2013 02:00:00 ==> 5
.
.
.
04/03/2013 21:00:00 ==> 26
04/03/2013 22:00:00 ==> 0
04/03/2013 23:00:00 ==> 9
Remove-LogOperation

Oct 10, 2013
Deletes configuration logs

Syntax

Remove-LogOperation -DatabaseCredentials <PSCredential> [-StartDateRange <DateTime>] [-EndDateRange <DateTime>] [-IncludeIncomplete] [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Remove-LogOperation -UserName <String> -SecurePassword <SecureString> [-StartDateRange <DateTime>] [-EndDateRange <DateTime>] [-IncludeIncomplete] [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]


Detailed Description

Remove-LogOperation deletes logs from the Configuration Logging database. This cmdlet targets high level operation logs for deletion. The associated low level operations logs are deleted as part of this operation via cascade deletion functionality present in the configuration logging database schema.

Related topics

Parameters

- **DatabaseCredentials** <PSCredential>

  Specifies the credentials of a database user with permission to delete records from the Configuration Logging database.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **UserName** <String>

  Specifies the user name of a database user with permission to delete records from the Configuration Logging database.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **SecurePassword** <SecureString>

  Specifies the password a database user, in secure string form, with permission to delete records from the Configuration Logging database.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **Password** <String>

  Specifies the password of a database user with permission to delete records from the Configuration Logging database.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **StartDateRange** <DateTime>

  Specifies the start date range for the deletion of logs.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
Specifies the start time of the earliest high level operation to delete

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>DateTime.Min</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-EndDateRange<DateTime>

Specifies the end time of the latest high level operation to delete

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>DateTime.UtcNow</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Incomplete<SwitchParameter>

Specifies if incomplete high level operations should be deleted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Remove-LogOperation -UserName "DOMAIN\User" -Password "UserPassword"
Remove all completed operation logs

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Remove-LogOperation -UserName "DOMAIN\User" -Password "UserPassword" -IncludeIncomplete
Remove all operations

-------------------------- EXAMPLE 3 --------------------------
C:\PS> Remove-LogOperation -username "domain\userName" -password "password" -StartDateRange "2013-01-01 12:00:00" -EndDateRange "2013-01-31 12:00:00"
Delete logs started on or after "2013-01-01 12:00:00" and completed on or before "2013-01-31 12:00:00"

-------------------------- EXAMPLE 4 --------------------------
Delete logs by supplying database user credentials via a credentials object.

C:\PS> `$securePassword = ConvertTo-SecureString -String "UserPassword" -AsPlainText -Force
C:\PS> $credentials = New-Object -TypeName System.Management.Automation.PSCredential -ArgumentList "DOMAIN\UserName", $securePassword
C:\PS> Remove-LogOperation -StartDateRange -DatabaseCredentials $credentials
Remove-LogServiceMetadata

Oct 10, 2013
Removes metadata from the given Service.

Syntax

Remove-LogServiceMetadata [-ServiceHostId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given Service.

Related topics

Set-LogServiceMetadata

Parameters

- **ServiceHostId**<Guid>
  
  Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **InputObject**<Service[]>
  
  Objects to which metadata is to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name**<String>
The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-Map<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-LoggingId<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

Error Codes

---------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-LogService | % { Remove-LogServiceMetadata -Map $_.MetadataMap }
Remove all metadata from all Service objects.
Remove-LogSiteMetadata

Oct 10, 2013
Removes metadata from the given Site.

Syntax

Remove-LogSiteMetadata -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-LogSiteMetadata -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-LogSiteMetadata -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-LogSiteMetadata -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given Site.

Related topics

Set-LogSiteMetadata

Parameters

-Name <String>
The metadata property to remove.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-Map <PSObject>
Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-
LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress**: <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

Error Codes

--------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-LogSite | % { Remove-LogSiteMetadata -Map $_.MetadataMap }
Remove all metadata from all Site objects.
Reset-LogDataStore

Oct 10, 2013
Refreshes the database string currently being used by the Log service.

Syntax

Reset-LogDataStore [-DataStore] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns the string for the database connection currently being used by the ConfigurationLogging Service. Can only be called for secondary data stores.

There is no requirement for a database connection to be configured in order for this command to be used.

Related topics

- Get-LogDataStore

Parameters

- **-DataStore**<String>

Specifies the database connection logical name to be used by the ConfigurationLogging Service. Can be either be 'Site' or the logical name of the secondary data store. Specifying the site data store will display an error because this operation is not supported for site data stores.

| Required? | true |
| Default Value | Site |
| Accept Pipeline Input? | false |

- **-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.ConfigurationLogging.Sdk.ServiceStatus

The status of the specified data store.

Notes

If the command fails, the following errors can be returned.

Error Codes

---------

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
EXAMPLE 1

c:\PS>Reset-LogDataStore -DataStore Secondary
OK

Refresh the database connection string for the ConfigurationLogging Service.
Reset-LogServiceGroupMembership

Oct 10, 2013
Reloads the access permissions and configuration service locations for the ConfigurationLogging Service.

Syntax

Reset-LogServiceGroupMembership [-ConfigServiceInstance] <ServiceInstance[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Enables you to reload ConfigurationLogging Service access permissions and configuration service locations. The Reset-LogServiceGroupMembership command must be run on at least one instance of the service type (Log) after installation and registration with the configuration service. Without this operation, the ConfigurationLogging services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-LogServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Related topics

Parameters

- **-ConfigServiceInstance** <ServiceInstance[]>

Specifies the configuration service instance object that represents the service instance for the type 'InterService' that references a configuration service for the deployment.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

- **-AdminAddress** <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Input Type

- **https://docs.citrix.com** © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1751
Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-LogServiceGroupMembership command.

Notes
If the command fails, the following errors can be returned.

Error Codes
--------
NoSuitableServiceInstance
None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-LogServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config -AdminAddress OtherServer.example.com | Reset-LogServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.
Set-LogDBConnection

Oct 10, 2013
Configures a database connection for the ConfigurationLogging Service.

Syntax

[<CommonParameters>]

Detailed Description

Configures a connection to a database in which the ConfigurationLogging Service can store its state. The service will attempt to connect and start using the database immediately after the connection is configured. The database connection string is updated to the specified value regardless of whether it is valid or not. Specifying an invalid connection string prevents a service from functioning until the error is corrected.

After a connection is configured, you cannot alter it without first clearing it (by setting the connection to $null).

You do not need to configure a database connection to use this command.

Related topics

Get-LogServiceStatus
Get-LogDBConnection
Test-LogDBConnection

Parameters

-**DBConnection**<String>

Specifies the database connection string to be used by the ConfigurationLogging Service. Passing in $null will clear any existing database connection configured.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Force**<SwitchParameter>

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

  | Required? | false |
  | Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
  | Accept Pipeline Input? | false |

- **DataStore**<String>

  Specifies the logical name of the data store for the ConfigurationLogging Service. Can be either be ‘Site’ or the logical name of the secondary data store.

  | Required? | false |
  | Default Value | Site |
  | Accept Pipeline Input? | false |

### Return Values


The `Set-LogDBConnection` command returns an object containing the status of the ConfigurationLogging Service together with extra diagnostics information.

- **DBUnconfigured**

  The ConfigurationLogging Service does not have a database connection configured.

- **DBRejectedConnection**

  The database rejected the logon attempt from the ConfigurationLogging Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- **InvalidDBConfigured**

  The expected stored procedures are missing from the database. This may be because the ConfigurationLogging Service schema has not been added to the database.

- **DBNotFound**

  The specified database could not be located with the configured connection string.

- **DBNewerVersionThanService**

  The version of the ConfigurationLogging Service currently in use is incompatible with the version of the ConfigurationLogging Service schema on the database. Upgrade the ConfigurationLogging Service to a more recent version.

- **DBOlderVersionThanService**

  The version of the ConfigurationLogging Service schema on the database is incompatible with the version of the ConfigurationLogging Service currently in use. Upgrade the database schema to a more recent version.
DBVersionChangeInProgress
A database schema upgrade is currently in progress.

OK
The ConfigurationLogging Service is running and is connected to a database containing a valid schema.

Failed
The ConfigurationLogging Service has failed.

Unknown
The status of the ConfigurationLogging Service cannot be determined.

Notes
If the command fails, the following errors can be returned.

Error Codes
----------

InvalidDBConnectionString
The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured
There was already a database connection configured. After a configuration is set, it can only be set to $null.

DatabaseError
An error occurred in the service while attempting a database operation.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS> Set-LogDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Configures a database connection string for the ConfigurationLogging Service.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Set-LogDBConnection -DBConnection "Invalid Connection String"

Invalid database connection string format.
Configures an invalid database connection string for the ConfigurationLogging Service.
Set-LogServiceMetadata

Oct 10, 2013
Adds or updates metadata on the given Service.

Syntax

Set-LogServiceMetadata [-ServiceHostId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-LogServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-LogServiceMetadata [-InputObject] <Service[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-LogServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Allows you to store additional custom data against given Service objects.

Related topics
Remove-LogServiceMetadata

Parameters

- **ServiceHostId** <Guid>
  
  Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **InputObject** <Service[]>

  Objects to which metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name** <String>
Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/:*.?=<>|[]"'

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

- **Value**<String>

Specifies the value for the property.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

- **Map**<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

- **LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default.
---|---
Accept Pipeline Input? | false

Return Values


Set-LogServiceMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>
Specifies the name of the property.

Value <string>
Specifies the value for the property.

Notes
If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.
ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS> Set-LogServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Set-LogSite

Oct 10, 2013
Sets global configuration logging settings.

Syntax

```
[<CommonParameters>]
```

Detailed Description

This cmdlet sets the global configuration logging settings.

Related topics

Get-LogSite

Parameters

- **-State**<LoggingState>

Sets the state of configuration logging. Values can be:

- **Enabled** - state changes will be logged. If logging is unavailable, the state change will still be permitted.
- **Disabled** - state changes will not be logged.
- **Mandatory** - state change will be logged. If logging is unavailable, the state change will not be permitted.

When the state is set to Enabled or Mandatory, XenDesktop services will attempt to log operations (which perform configuration changes) before performing them.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Locale**<String>

Sets the language that logs should be recorded in. Values can be: 'en', 'ja', 'zh-CN', 'de', 'es' or 'fr'.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-PassThru**<SwitchParameter>

Returns the affected record. By default, this cmdlet does not generate any output.
### **-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>False</td>
</tr>
</tbody>
</table>

### **-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>False</td>
</tr>
</tbody>
</table>

Return Values

Citrix.ConfigurationLogging.Sdk.Site

Current logging settings

Notes

Configuration logging will automatically transition to a 'NotSupported' state if the logging feature is not licensed. Set-LogSite will reject request to set logging to 'Enabled' or 'Mandatory' while the logging feature is not licensed.

Examples

------------ EXAMPLE 1 --------------------------

C:\PS> Set-LogSite -Locale "zh-CN"
Set the logging language to Chinese. The logging state will be unchanged.

------------ EXAMPLE 2 --------------------------

C:\PS> Set-LogSite -State "Mandatory"
Set the logging state to mandatory. The logging locale will be unaffected. In this state, no configuration change will be
allowed unless it is successfully logged.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> Set-LogSite -Locale "de" -State "Enabled"

Set the logging language to German and the state to Enabled.
Set-LogSiteMetadata

Oct 10, 2013
Adds or updates metadata on the given Site.

Syntax

Set-LogSiteMetadata -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-LogSiteMetadata -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-LogSiteMetadata -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-LogSiteMetadata -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Provides the ability for additional custom data to be stored against given Site objects.

Related topics
Remove-LogSiteMetadata

Parameters

- **Name**: <String>
  Specifies the property name of the metadata to be added. The property must be unique for the Site specified. The property cannot contain any of the following characters /:.*?=<|[]"'
  
  | Required? | true |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **Value**: <String>
  Specifies the value for the property.
  
  | Required? | true |
  | Default Value | |
  | Accept Pipeline Input? | false |

- **Map**: <PSObject>
  Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").
  
  | Required? | true |
  | Default Value | |

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1764
Accept Pipeline Input? | true (ByValue)

- **LoggingId**<Guid>

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


Set-LogSiteMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>

Specifies the name of the property.

Value <string>

Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.
DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS> Set-LogSiteMetadata -SiteDUMMY_IdProperty 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Site with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Start-LogHighLevelOperation

Logs the start of a high level operation.

Syntax


Detailed Description

Start-LogHighLevelOperation creates a log entry to record the start of a high level operation.

Start-LogHighLevelOperation can be called to log high level configuration changes which originate from customized configuration scripts. Start-LogHighLevelOperation should be called before the script invokes service SDK cmdlets which execute the configuration changes.

Start-LogHighLevelOperation returns a HighLevelOperation object which contains information about the started high level operation. The "Id" property of the returned HighLevelOperation uniquely identifies the started high level operation. This can be supplied into the "-LoggingId" parameter which is implemented in all service SDK cmdlets which execute loggable configuration changes.

High level operation logs are automatically created by the XenDesktop consoles when:

- Initiating an operation which performs configuration changes.
- Initiating an operation which performs an administration activity.

Configuration Change and Administrator Activity

A configuration change is an operation which alters the configuration of the XenDesktop site. Examples of configuration changes include:

- Creating or editing a host.
- Creating or editing a catalog.
- Adding a user to a delivery group.
- Deleting a machine.

An administrator activity operation doesn’t directly alter site configuration, but it could be an operation carried out by an administrator as part of site management or helpdesk support. Examples of administrator activities include:

- Shutdown/start/restart of a user desktop.
- Studio or Director sending a message to a user.
- Rebooting a user’s desktop.

Once the change being logged has completed (whether successfully or not) the Stop-LogHighLevelOperation cmdlet should be called to log the completion of a started high level operation.

Related topics

Stop-LogHighLevelOperation
Get-LogHighLevelOperation

Parameters

- **-Text**: <String>

  Specifies text to describe the high level operation.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Source**: <String>

  Specifies the source of the high level operation.
Required? | true
---|---
Default Value | 
Accept Pipeline Input? | false

**-StartTime**<br>`<DateTime>`

Specifies the start time of the high level operation.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td><code>DateTime.UtcNow</code></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-OperationType**<br>`<OperationType>`

Specifies the type of operation being logged. Values can be:
- AdminActivity - If the operation being logged performs an administration activity.
- ConfigurationChange - If the operation being logged performs a configuration change.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td><code>ConfigurationChange</code></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-TargetTypes**<br>`<String[]>`

Specifies the names of the object types that will be affected by the operation being logged.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Parameters**<br>`<Hashtable>`

Specifies the names and values of parameters that are supplied to the operation being logged.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<br>`<String>`

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td><code>Localhost</code>. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

`Citrix.ConfigurationLogging.Sdk.HighLevelOperation`

The newly logged high level operation start.

**Examples**

```
C:\PS> $succeeded = $false # indicates if high level operation succeeded.
C:\PS> # Log high level operation start.
C:\PS> $highLevelOp = Start-LogHighLevelOperation -Text "Create catalog" -Source "My Custom Script"
```

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1768
C:\PS> try
C:\PS> { # Create catalog object
C:\PS>   $catalog = New-BrokerCatalog -Name "MyCatalog" -ProvisioningType Manual -AllocationType Permanent -MinimumFunctionalLevel 'LMAX' -LoggingId $highLevelOp.Id
C:\PS> 
C:\PS>   # Add a machine to the catalog
C:\PS>   $machine = New-BrokerMachine -CatalogUid $catalog.Uid -MachineName "DOMAIN\Machine" -LoggingId $highLevelOp.Id
C:\PS>   $succeeded = $true
C:\PS> }
C:\PS> catch{ "Error encountered" }
C:\PS> finally{
C:\PS>   # Log high level operation stop, and indicate its success
C:\PS>   Stop-LogHighLevelOperation -HighLevelOperationId $highLevelOp.Id -IsSuccessful $succeeded
C:\PS> }

Creates a catalog and assigns a machine to it, within the scope of a high level operation start and stop. The identifier of the high level operation is passed into the
"-LoggingId" parameter of the service SDK cmdlets. The execution of the cmdlets in the services will create the low level operation logs for the supplied high level operation.
Stop-LogHighLevelOperation

Logs the completion of a previously started high level operation.

Syntax

Stop-LogHighLevelOperation -HighLevelOperationId <String> -IsSuccessful <Boolean> [-EndTime <DateTime>] [-AdminAddress <String>] [-<CommonParameters>] 

Detailed Description

Stop-LogHighLevelOperation logs the completion of a started high level operation.

Related topics

Start-LogHighLevelOperation
Get-LogHighLevelOperation

Parameters

- **HighLevelOperationId**:<String>

- Specifies the identifier of the high level operation being completed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

- **IsSuccessful**:<Boolean>

- Specifies if the started high level operation completed successfully.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

- **EndTime**:<DateTime>

- Specifies the end time of the high level operation being completed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>DateTime.UtcNow.</td>
</tr>
</tbody>
</table>

- **AdminAddress**:<String>

- Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Examples

-------------------------- EXAMPLE 1 --------------------------

```powershell
C:\PS> $succeeded = $false #indicates if high level operation succeeded.
C:\PS> # Log high level operation start.
C:\PS> $highLevelOp = Start-LogHighLevelOperation -Text "Create unmanaged catalog" -Source "My Custom Script"
C:\PS> try
C:\PS> {
C:\PS>   # Create catalog object
```
C:\PS> $catalog = New-BrokerCatalog -Name "MyCatalog" -ProvisioningType Manual -AllocationType Permanent -MinimumFunctionalLevel 'LMAX' -LoggingId $highLevelOp.Id
C:\PS>
C:\PS> # Add a machine to the catalog
C:\PS> $machine = New-BrokerMachine -CatalogUid $catalog.Uid -MachineName "DOMAIN\Machine" -LoggingId $highLevelOp.Id
C:\PS> $succeeded = $true
C:\PS> }
C:\PS> catch{ "Error encountered" }
C:\PS>
C:\PS> finally{
C:\PS>   # Log high level operation stop, and indicate its success
C:\PS>   Stop-LogHighLevelOperation -HighLevelOperationId $highLevelOp.Id -IsSuccessful $succeeded
C:\PS>   }

Creates an unmanaged catalog and assigns a machine to it, within the scope of a high level operation start and stop. The identifier of the high level operation is passed into the 
"-LoggingId" parameter of the service SDK cmdlets. The execution of the cmdlets in the services will create the low level operation logs for the supplied high level operation.
Test-LogDBConnection

Oct 10, 2013
Tests a database connection for the ConfigurationLogging Service.

Syntax

Test-LogDBConnection [-DBConnection] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-DataStore <String>] [CommonParameters]

Detailed Description

Tests a connection to the database in which the ConfigurationLogging Service can store its state. The service will attempt to connect to the database without affecting the current connection to the database.

You do not have to clear the connection to use this command.

Related topics

Get-LogServiceStatus
Get-LogDBConnection
Set-LogDBConnection

Parameters

- **DBConnection**:<String>

  Specifies the database connection string to be tested by the ConfigurationLogging Service.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**:<Guid>

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**:<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

- **DataStore**:<String>
Specifies the logical name of the data store for the ConfigurationLogging Service. Can be either be ‘Site’ or the logical name of the secondary data store.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Site</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


The Test-LogDBConnection command returns an object containing the status of the ConfigurationLogging Service if the connection string of the specified data store were to be set to the string being tested, together with extra diagnostics information for the specified connection string.

DBRejectedConnection

The database rejected the logon attempt from the ConfigurationLogging Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the ConfigurationLogging Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBNewerVersionThanService

The version of the ConfigurationLogging Service currently in use is incompatible with the version of the ConfigurationLogging Service schema on the database. Upgrade the ConfigurationLogging Service to a more recent version.

DBOlderVersionThanService

The version of the ConfigurationLogging Service schema on the database is incompatible with the version of the ConfigurationLogging Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Set-LogDBConnection command would succeed if it were executed with the supplied connection string.

Failed

The ConfigurationLogging Service has failed.

Unknown

The status of the ConfigurationLogging Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes
InvalidDBConnectionString
The database connection string has an invalid format.

DatabaseError
An error occurred in the service while attempting a database operation.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Test-LogDBConnection -DBConnection "Server=serverName;SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Tests a database connection string for the ConfigurationLogging Service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Test-LogDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Tests an invalid database connection string for the ConfigurationLogging Service.
## Citrix.DelegatedAdmin.Admin.V1

**Oct 10, 2013**

### Overview

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdminDelegatedAdminSnapin</td>
<td>The Delegated Administration Service PowerShell snap-in provides</td>
</tr>
<tr>
<td>Admin Filtering</td>
<td>Describes the common filtering options for XenDesktop cmdlets.</td>
</tr>
</tbody>
</table>

### Cmdlets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-AdminPermission</td>
<td>Add permissions to the set of permissions of a role.</td>
</tr>
<tr>
<td>Add-AdminRight</td>
<td>Grants a given right to the specified administrator.</td>
</tr>
<tr>
<td>Get-AdminAdministrator</td>
<td>Gets administrators configured for this site.</td>
</tr>
<tr>
<td>Get-AdminDBConnection</td>
<td>Gets the database string for the specified data store used by the DelegatedAdmin Service.</td>
</tr>
<tr>
<td>Get-AdminDBSchema</td>
<td>Gets a script that creates the DelegatedAdmin Service database schema for the specified data store.</td>
</tr>
<tr>
<td>Get-AdminDBVersionChangeScript</td>
<td>Gets a script that updates the DelegatedAdmin Service database schema.</td>
</tr>
<tr>
<td>Get-AdminEffectiveAdministrator</td>
<td>Retrieve the effective administrator objects for a user.</td>
</tr>
<tr>
<td>Get-AdminEffectiveRight</td>
<td>Gets the set of Right objects associated with the current user.</td>
</tr>
<tr>
<td>Get-AdminInstalledDBVersion</td>
<td>Gets a list of all available database schema versions for the DelegatedAdmin Service.</td>
</tr>
<tr>
<td>Get-AdminPermission</td>
<td>Gets permissions configured for the site.</td>
</tr>
<tr>
<td>Get-AdminPermissionGroup</td>
<td>Gets permission groups configured for the site.</td>
</tr>
<tr>
<td>Get-AdminRevision</td>
<td>Gets the current revision of the delegated administration configuration data.</td>
</tr>
<tr>
<td>Get-AdminRole</td>
<td>Gets roles configured for this site.</td>
</tr>
<tr>
<td>Get-AdminRoleConfiguration</td>
<td>Gets role configurations for this site.</td>
</tr>
<tr>
<td>Get-AdminScope</td>
<td>Gets scopes configured for this site.</td>
</tr>
<tr>
<td>Get-AdminService</td>
<td>Gets the service record entries for the DelegatedAdmin Service.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AdminServiceAddedCapability</td>
<td>Added capabilities for the DelegatedAdmin Service on the controller.</td>
</tr>
<tr>
<td>Get-AdminServiceInstance</td>
<td>Gets the service instance entries for the DelegatedAdmin Service.</td>
</tr>
<tr>
<td>Get-AdminServiceStatus</td>
<td>Gets the current status of the DelegatedAdmin Service on the controller.</td>
</tr>
<tr>
<td>Import-AdminRoleConfiguration</td>
<td>Imports role configuration data into the Delegated Administration Service.</td>
</tr>
<tr>
<td>New-AdminAdministrator</td>
<td>Adds a new administrator to the site.</td>
</tr>
<tr>
<td>New-AdminRole</td>
<td>Adds a new custom role to the site.</td>
</tr>
<tr>
<td>New-AdminScope</td>
<td>Adds a new scope to the site.</td>
</tr>
<tr>
<td>Remove-AdminAdministrator</td>
<td>Removes administrators from the site.</td>
</tr>
<tr>
<td>Remove-AdminAdministratorMetadata</td>
<td>Removes metadata from the given Administrator.</td>
</tr>
<tr>
<td>Remove-AdminPermission</td>
<td>Remove permissions from the set of permissions of a role.</td>
</tr>
<tr>
<td>Remove-AdminRight</td>
<td>Removes rights from an administrator.</td>
</tr>
<tr>
<td>Remove-AdminRole</td>
<td>Removes a role from the site.</td>
</tr>
<tr>
<td>Remove-AdminRoleMetadata</td>
<td>Removes metadata from the given Role.</td>
</tr>
<tr>
<td>Remove-AdminScope</td>
<td>Removes a scope from the site.</td>
</tr>
<tr>
<td>Remove-AdminScopeMetadata</td>
<td>Removes metadata from the given Scope.</td>
</tr>
<tr>
<td>Remove-AdminServiceMetadata</td>
<td>Removes metadata from the given Service.</td>
</tr>
<tr>
<td>Rename-AdminRole</td>
<td>Rename a role</td>
</tr>
<tr>
<td>Rename-AdminScope</td>
<td>Rename a scope</td>
</tr>
<tr>
<td>Reset-AdminServiceGroupMembership</td>
<td>Reloads the access permissions and configuration service locations for the DelegatedAdmin Service.</td>
</tr>
<tr>
<td>Set-AdminAdministrator</td>
<td>Sets the properties of an administrator.</td>
</tr>
<tr>
<td>Set-AdminAdministratorMetadata</td>
<td>Adds or updates metadata on the given Administrator.</td>
</tr>
<tr>
<td>Set-AdminDBConnection</td>
<td>Configures a database connection for the DelegatedAdmin Service.</td>
</tr>
<tr>
<td>Set-AdminRole</td>
<td>Set the properties of a role.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>Set-AdminRoleMetadata</td>
<td>Adds or updates metadata on the given Role.</td>
</tr>
<tr>
<td>Set-AdminScope</td>
<td>Set the properties of a scope.</td>
</tr>
<tr>
<td>Set-AdminScopeMetadata</td>
<td>Adds or updates metadata on the given Scope.</td>
</tr>
<tr>
<td>Set-AdminServiceMetadata</td>
<td>Adds or updates metadata on the given Service.</td>
</tr>
<tr>
<td>Test-AdminAccess</td>
<td>Retrieves the scopes where the specified operation is permitted.</td>
</tr>
<tr>
<td>Test-AdminDBConnection</td>
<td>Tests a database connection for the DelegatedAdmin Service.</td>
</tr>
</tbody>
</table>
about_AdminDelegatedAdminSnapin

Oct 10, 2013

TOPIC

about_AdminDelegatedAdminSnapin

SHORT DESCRIPTION

The Delegated Administration Service PowerShell snap-in provides administrative functions for the Delegated Administration Service.

COMMAND PREFIX

All commands in this snap-in are prefixed with 'Admin'.

LONG DESCRIPTION

The Delegated Administration Service PowerShell snap-in enables both local and remote administration of the Delegated Administration Service.

The Delegated Administration Service (or DAS for short) stores information about Citrix administrators and the rights they have. Services in the XenDesktop deployment use the DAS to determine whether a particular user has the privilege to perform an operation or not.

The snap-in provides storage and configuration of these entities:

Administrators

Each administrator object represents an individual person or a group of people identified by their Active Directory account. Administrators can be enabled and disabled.

The effective rights that a user has is the superset of any rights that they have by looking at their Active Directory group membership. Disabled administrator entries are ignored for this calculation.

Once a site is setup, there must always be a full administrator and the Delegated Administration snap-in rejects requests to remove or disable the last full administrator.

Roles

A role represents a job function. That is, anyone with a given role is expected to be able to use or perform the tasks, wizards, and actions associated with that role. Administrators may have multiple roles for a particular site.
Some roles are built-in, and some editions of the product allow custom roles to be created with different combinations of permissions.

Scopes

Scopes represent a collection of objects, and are used to group objects for administrative purposes in a way that is relevant to the organisation. They can be used to represent both hierarchical and non-hierarchical relationships.

Objects can exist in multiple scopes at once. You may find it easier to think of scopes as labels, or a non-exclusive grouping such as a play-list.

All objects are implicitly in the built-in 'All' scope.

Some objects are not scoped, and access to them is through either the 'All' scope or indirectly through a scoped object. For example sessions are not directly scoped but can be accessed using the scope of the desktop group.

The DAS stores information about scopes, but the mapping between scopes and objects is stored and updated using the PowerShell snap-ins of each corresponding service. For example, Delivery Group scopes are managed using the Broker PowerShell snap-in.

Rights

Rights determine what an administrator can do and where they can do it. They are expressed as a number of <role, scope> pairs associated with each administrator.

To gain access to any particular object, a person must match an administrator object that has an appropriate right that allows the required operation in a scope that the object is a member of.

Permissions

Each task, wizard or action in the Citrix Studio or Director consoles represents a unit of functionality that an administrator can perform. Permissions are expressed at a high level and generally correspond directly to the labels in the consoles. For example: "Edit catalog", or "Create delivery group".
Permission groups:

Permissions are grouped into related functionality when displayed by the console.

Operations

Operations are the indivisible unit of functionality that each XenDesktop service can perform, and usually correspond to individual cmdlets. Internally, each permission requires a number of operations to be performed, possibly by different services.
about_Admin_Filtering

Oct 10, 2013

TOPIC

XenDesktop - Advanced Dataset Filtering

SHORT DESCRIPTION

Describes the common filtering options for XenDesktop cmdlets.

LONG DESCRIPTION

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

```
Get-<Noun> | Where { $_.Size = 'Small' } | Sort 'Date' | Select -First 10
```

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

```
Get-<Noun> -Size 'Small' -SortBy 'Date' -MaxRecordCount 10
```

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

- `-MaxRecordCount <int>`

  Specifies the maximum number of results to return.
  For example, to return only the first nine results use:

  ```
  Get-<Noun> -MaxRecordCount 9
  ```

  If not specified, only the first 250 records are returned, and if more are available, a warning is produced:
WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

To retrieve all records, specify a large number for -MaxRecordCount. As the value is an integer, you can use the following:

Get-<Noun> -MaxRecordCount [int]:MaxValue

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....

Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<< -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (...) [Get-<Noun>], PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.
Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values. Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

-SortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter 'Name -like "High*" -or (Priority -eq 1 -and Severity -ge 2)'

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

EXAMPLES

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company "citrix" -Product '[X]EN''
Get-<Noun> -Product "Xen*" -Company "CITRIX"
Get-<Noun> -Filter { Company -eq 'Citrix' -and Product -like 'Xen*'}
See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
Get-<Noun> -Company "Abc[*]"                 # Matches Abc*
Get-<Noun> -Company "Abc`*"                  # Matches Abc*
Get-<Noun> -Filter { Company -eq "Abc`*" }   # Matches Abc*
Get-<Noun> -Filter { Company -eq "A`"B`C" }   # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

```
Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```
Get-<Noun> -Filter 'Enabled'      # Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled' # Equivalent to 'Enabled -eq $false'
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```
Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:
```powershell
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle" }
Get-<Noun> -Filter { Shape -like 'C' }
```

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

```powershell
Get-<Noun> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z" }
$d = [DateTime]"2010-08-23T12:30:00.0Z"
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }
```

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

```powershell
Get-<Noun> -Filter { StartTime -ge '-2' }          # Two days ago
Get-<Noun> -Filter { StartTime -ge '-1:30' }       # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge '-0:0:30' }     # 30 seconds ago
```

**ARRAY PROPERTIES**

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings. Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

```powershell
Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*" }
Get-<Noun> -Filter { Users -contains "Fred*" }
```

You can also use the singular form with -Filter to search using other operators:

```powershell
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter { User -eq "Frederick" }
# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter { User -lt 'F' }
```

**COMPLEX EXPRESSIONS**
When matching against multiple values, you can use a sequence of
comparisons joined with -or operators, or you can use -in and -notin:

```
Get-<Noun> -Filter { Shape -eq 'Circle' -or Shape -eq 'Square' }
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }
```

Braces can be used to group complex expressions, and override the default
left-to-right evaluation of -and and -or. You can also use -not to invert
the sense of any sub-expression:

```
Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue' -and Shape -eq 'Circle')} 
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue' -and Shape -eq 'Circle')} 
```

**PAGING**
The simplest way to page through data is to use the -Skip and
-MaxRecordCount parameters. So, to read the first three pages of data with
10 records per page, use:

```
Get-<Noun> -Skip 0  -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 10 -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 20 -MaxRecordCount 10 <other filtering criteria>
```

You must include the same filtering criteria on each call, and
ensure that the data is sorted consistently.

The above approach is often acceptable, but as each call performs an
independent query, data changes can result in records being skipped or
appearing twice. One approach to improve this is to sort by a unique id
field and then start the search for the next page at the unique id after
the last unique id of the previous page. For example:

```
# Get the first page
Get-<Noun> -MaxRecordCount 10 -SortBy SerialNumber

<table>
<thead>
<tr>
<th>SerialNumber</th>
</tr>
</thead>
<tbody>
<tr>
<td>A120004</td>
</tr>
<tr>
<td>A120007</td>
</tr>
<tr>
<td>... 7 other records ...</td>
</tr>
<tr>
<td>A120900</td>
</tr>
</tbody>
</table>

# Get the next page
Get-<Noun> -MaxRecordCount 10 -Filter { FirstName -gt 'A120900' }

<table>
<thead>
<tr>
<th>SerialNumber</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
```
FILTER SYNTAX DEFINITION

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= "{" <ComponentList> "}"

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |
                   <Component>

<Component> ::= <NotOperator> <Factor> |

<Factor> ::= "{" <ComponentList> "}" |
          <PropertyName> <ComparisonOperator> <Value> |
          <PropertyName>

<AndOrOperator> ::= "-and" | "-or"

<NotOperator> ::= "-not" | "!

<ComparisonOperator> ::= "-eq" | "-ne" | "-le" | "-ge" | "-lt" | "-gt" |
                       "-like" | "-notlike" | "-contains" | "-notcontains" |
                       "-in" | "-notin"

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |
          <scalar variable> | <array variable> |
          "$null" | "$true" | "$false"
Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30" means 1 hour and 30 minutes ago.
Add-AdminPermission

Oct 10, 2013
Add permissions to the set of permissions of a role.

Syntax

Add-AdminPermission [-InputObject] <Permission[]> -Role <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Add-AdminPermission [-Permission] <String[]> -Role <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Add extra permissions to the set of permissions that a role maps to.

Any administrator with a right including that role immediately gains the ability to use the operations of the new permissions.

Duplicate permissions do not produce an error, and permissions are skipped if the role already contains the permission (without error).

You cannot modify the permissions of built-in roles.

Related topics

Remove-AdminPermission
Get-AdminPermission
Get-AdminRole
Get-AdminPermissionGroup
Test-AdminAccess

Parameters

-InputObject <Permission[]>

Specifies the permissions to add.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Permission <String[]>

Specifies the list of permissions to add (by identifier).
### Required?

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-Role**<String>

Role name or identifier of the role to update.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.DelegatedAdmin.Sdk.Permission You can pipe a list of permissions to be added into this command.

**Return Values**

None

**Examples**

-------------------------- EXAMPLE 1 --------------------------
Add-AdminPermission -Role MyRole -Permission Global_Read,Logging_Read
Add a couple of specific permissions to the 'MyRole' role.

-------------------------- EXAMPLE 2 --------------------------

Get-AdminRole "Delivery Administrator" | Select -Expand Permissions
Add all of the permissions of the Delivery Administrator role to MyRole.
Add-AdminRight

Oct 10, 2013
Grants a given right to the specified administrator.

Syntax

Add-AdminRight -Scope <String> -Role <String> -Administrator <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]
Add-AdminRight -Role <String> -Administrator <String> [-All] [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

Use the Add-AdminRight cmdlet to add rights (role and scope pairs) to an administrator.

For convenience, you can use the -All parameter to specify the 'All' scope.

Use the Get-AdminAdministrator cmdlet to determine what rights an administrator has.

Related topics

Get-AdminAdministrator
Get-AdminEffectiveRight
Remove-AdminRight

Parameters

-InputObject <Right[]>

Specifies the rights to add from Right objects.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-Scope <String>

Specifies the scope name or scope identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
### Accept Pipeline Input?
| Accept Pipeline Input? | false |

### Role <String>
Specifies the role name or role identifier.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

### Administrator <String>
Specifies the name or SID of the administrator.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

### All <SwitchParameter>
Specifies the 'All' scope. This parameter avoids localization issues or having to type the identifier of the 'All' scope.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<p>| Required? | |
| Default Value | |
| Accept Pipeline Input? | false |</p>
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.DelegatedAdmin.Sdk.Right You can pipe the rights to be added into this command.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Add-AdminRight -Role 'Help Desk Administrator' -Scope London -Administrator DOMAIN\Admin1
Assigns the 'Help Desk Administrator' role and 'London' scope to the 'Admin1' administrator.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Add-AdminRight -Role 'Full Administrator' -All -Administrator DOMAIN\Admin1
Assigns the 'Full Administrator' role and 'All' scope to the 'Admin1' administrator.

-------------------------- EXAMPLE 3 --------------------------

C:\PS> $admin = Get-AdminAdministrator -Name DOMAIN\ExistingAdmin
C:\PS> Add-AdminRight -InputObject $admin.Rights -Administrator DOMAIN\NewAdmin
Copies the administrator rights from 'ExistingAdmin' to 'NewAdmin'.
Get-AdminAdministrator

Oct 10, 2013
Gets administrators configured for this site.

Syntax


Detailed Description

Retrieves administrators matching the specified criteria. If no parameters are specified this cmdlet enumerates all administrators.

See about_Admin_Filtering for information about advanced filtering options.

Related topics

New-AdminAdministrator
Set-AdminAdministrator
Remove-AdminAdministrator
Set-AdminAdministratorMetadata
Remove-AdminAdministratorMetadata

Parameters

- **Name**:<String>

  Gets administrators with the specified name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **Sid**:<String>

  Gets administrators with the specified SID (security identifier).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
-**Enabled**<Boolean>

Gets administrators with the specified value of Enabled.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Metadata**<String>

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**ReturnTotalRecordCount**<SwitchParameter>

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**MaxRecordCount**<Int32>

Specifies the maximum number of records to return.
| Required? | false |
| Default Value | 250 |
| Accept Pipeline Input? | false |

**-Skip<Int32>**

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | false |

**-SortBy<String>**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |
| Accept Pipeline Input? | false |

**-Filter<String>**

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |
Input Type
None You cannot pipe input into this cmdlet.

Return Values
Citrix.DelegatedAdmin.Sdk.Administrator

Get-AdminAdministrator returns an object for each matching administrator.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-AdminAdministrator -Name DOMAIN\TestUser
Finds the administrator object (if one exists) for user "DOMAIN\TestUser".

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-AdminAdministrator -Enabled $false
Finds all disabled administrator objects.
Get-AdminDBConnection

Oct 10, 2013
Gets the database string for the specified data store used by the DelegatedAdmin Service.

Syntax
Get-AdminDBConnection [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Returns the database connection string for the specified data store.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

Related topics
Get-AdminServiceStatus
Set-AdminDBConnection
Test-AdminDBConnection

Parameters
-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
system:string
The database connection string configured for the DelegatedAdmin Service.

Notes
If the command fails, the following errors can be returned.

Error Codes
----------
NoDBConnections
The database connection string for the DelegatedAdmin Service has not been specified.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> Get-AdminDBConnection

Server=serverName\SQLExpress;Initial Catalog = databaseName;  Integrated Security = True
Get the database connection string for the DelegatedAdmin Service.
Get-AdminDBSchema

Oct 10, 2013

Gets a script that creates the DelegatedAdmin Service database schema for the specified data store.

Syntax

Get-AdminDBSchema [-DatabaseName <String>] [-ServiceGroupName <String>] [-ScriptType <ScriptTypes>] [-LocalDatabase] [-Sid <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Gets SQL scripts that can be used to create a new DelegatedAdmin Service database schema, add a new DelegatedAdmin Service to an existing site, remove a DelegatedAdmin Service from a site, or create a database server logon for a DelegatedAdmin Service. If no Sid parameter is provided, the scripts obtained relate to the currently selected DelegatedAdmin Service instance, otherwise the scripts relate to DelegatedAdmin Service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied. The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to a DelegatedAdmin SDK cmdlet. The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured. The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SMDCMD mode'. The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to DelegatedAdmin Service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to DelegatedAdmin Service roles

If ScriptType is Evict, the returned script contains:

- Removal of DelegatedAdmin Service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

If the service uses two data stores they can exist in the same database. You do not need to configure a database before
using this command.

Related topics

Set-AdminDBConnection

Parameters

**-DatabaseName**<String>

Specifies the name of the database for which the schema will be generated.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-ServiceGroupName**<String>

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the DelegatedAdmin services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

**-ScriptType**<ScriptTypes>

Specifies the type of database script returned. Available script types are:

**Database**

Returns a full database script that can be used to create a database schema for the DelegatedAdmin Service in a database instance that does not already contain a schema for this service. The DatabaseName and ServiceGroupName parameters must be specified to create a script of this type.

**Instance**

Returns a permissions script that can be used to add further DelegatedAdmin services to an existing database instance that already contains the full DelegatedAdmin service schema, associating the services to the Service Group. The Sid parameter can optionally be specified to create a script of this type.

**Login**

Returns a database logon script that can be used to add the required logon accounts to an existing database instance that contains the DelegatedAdmin Service schema. This is used primarily when creating a mirrored database environment. The DatabaseName parameter must be specified to create a script of this type.
Evict

Returns a script that can be used to remove the specified DelegatedAdmin Service from the database entirely. The DatabaseName and Sid parameters must be specified to create a script of this type.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>DatabaseName</td>
<td>false</td>
</tr>
<tr>
<td>Sid</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LocalDatabase**<SwitchParameter>

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for DelegatedAdmin services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Sid**<String>

Specifies the SID of the controller on which the DelegatedAdmin Service instance to remove from the database is running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.String

A string containing the required SQL script for application to a database.
Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

Create the database schema.
Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:
Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:
Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned.

Error Codes
-----------

GetSchemasFailed
The database schema could not be found.

ActiveDirectoryAccountResolutionFailed
The specified Active Directory account or Group could not be found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AdminDBSchema -DatabaseName MyDB -ServiceGroupName MyServiceGroup > c:\AdminSchema.sql
Get the full database schema for site data store of the DelegatedAdmin Service and copy it to a file called 'c:\AdminSchema.sql'.

This script can then be used to create the schema in a pre-existing database named 'MyDB' that does not already contain a DelegatedAdmin Service site schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-AdminDBSchema -DatabaseName MyDB -scriptType Login > c:\DelegatedAdminLogins.sql
Get the logon scripts for the DelegatedAdmin Service.
Get-AdminDBVersionChangeScript

Oct 10, 2013

Gets a script that updates the DelegatedAdmin Service database schema.

Syntax


Detailed Description

Returns a database script that can be used to upgrade or downgrade the site or secondary schema for the DelegatedAdmin Service from the current schema version to a different version.

Related topics

Get-AdminInstalledDBVersion

Parameters

- **DatabaseName (String)**

  Specifies the name of the database instance to which the update applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **TargetVersion (Version)**

  Specifies the version of the database you want to update to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress (String)**

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values
System.Management.Automation.PSObject
A PSObject containing the required SQL script for application to a database.

Notes
The PSObject returned by this cmdlet contains the following properties:

-- Script The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

-- NeedExclusiveAccess Indicates whether all services in the service group must be shut down during the update or not.

-- CanUndo Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any DelegatedAdmin services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-AdminServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned.

Error Codes
----------
NoOp
The operation was successful but had no effect.

NoDBConnections
The database connection string for the DelegatedAdmin Service has not been specified.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $update = Get-AdminDBVersionChangeScript -DatabaseName MyDb -TargetVersion 1.0.75.0
C:\PS> $update.Script > update_75.sql
Gets an SQL update script to update the current schema to version 1.0.75.0. The resulting update_75.sql script is suitable for direct use with the SQL Server SQLCMD utility.
Get-AdminEffectiveAdministrator

Oct 10, 2013
Retrieve the effective administrator objects for a user.

Syntax

Get-AdminEffectiveAdministrator [-Name] <String> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

This command determines what groups the specified user belongs to and retrieves the matching administrator records. It includes the set of rights that would be granted to the user if he or she used the system.

As this command uses Active Directory to determine what groups the user has, the caller must have the ability to read this information from Active Directory.

Only enabled administrator records are returned.

Related topics

Get-AdminAdministrator

Parameters

-Name <String>
User name or SID of user to query

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

string User name or SID of user to query

Return Values
Citrix.DelegatedAdmin.Sdk.Administrator

Administrator records matching the specified user

Examples

---------------------------- EXAMPLE 1 ------------------------------

C:\PS> Get-AdminEffectiveAdministrator MYDOMAIN\testuser
Retrieve the administrator records matching user 'testuser'.
Get-AdminEffectiveRight

Oct 10, 2013
Gets the set of Right objects associated with the current user.

Syntax

Get-AdminEffectiveRight [-AdminAddress <String>] [CommonParameters>

Detailed Description

The Get-AdminEffectiveRight cmdlet returns the effective rights of the current user. This is the superset of all rights of the enabled administrators that the current user matches, taking into account Active Directory group membership.

Related topics

Get-AdminAdministrator
Add-AdminRight
Remove-AdminRight

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.DelegatedAdmin.Sdk.Right

The Rights associated with the current user

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-AdminEffectiveRight
Return the effective rights for the current user.
Get-AdminInstalledDBVersion

Oct 10, 2013

Gets a list of all available database schema versions for the DelegatedAdmin Service.

Syntax

Get-AdminInstalledDBVersion [-Upgrade] [-Downgrade] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns the current version of the DelegatedAdmin Service database schema, if no flags are set, otherwise returns versions for which upgrade or downgrade scripts are available and have been stored in the database.

Related topics

Parameters

- **-Upgrade**: <SwitchParameter>

Specifies that only schema versions to which the current database version can be updated should be returned.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

- **-Downgrade**: <SwitchParameter>

Specifies that only schema versions to which the current database version can be reverted should be returned.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

- **-AdminAddress**: <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values
System.Version

The Get-AdminInstalledDbVersion command returns objects containing the new definition of the DelegatedAdmin Service database schema version.

Major <Integer>
Minor <Integer>
Build <Integer>
Revision <Integer>

Notes
If the command fails, the following errors can be returned.

Error Codes
-----------
InvalidParameterCombination
Both the Upgrade and Downgrade flags were specified.
NoOp
The operation was successful but had no effect.
NoDBConnections
The database connection string for the DelegatedAdmin Service has not been specified.
DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.
PermissionDenied
You do not have permission to execute this command.
AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.
CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AdminInstalledDBVersion

<table>
<thead>
<tr>
<th>Major</th>
<th>Minor</th>
<th>Build</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Get the currently installed version of the DelegatedAdmin Service database schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-AdminInstalledDBVersion -Upgrade

<table>
<thead>
<tr>
<th>Major</th>
<th>Minor</th>
<th>Build</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Get the versions of the DelegatedAdmin Service database schema for which upgrade scripts are supplied.
Get-AdminPermission

Oct 10, 2013

Gets permissions configured for the site.

Syntax


Detailed Description

Retrieves permission matching the specified criteria. If no parameters are specified this cmdlet enumerates all permissions.

Permissions are configured using the Import-AdminRoleConfiguration command, and are used to represent collections of operations that are needed to perform a particular task. These permissions are presented in Citrix Studio when configuring roles.

Permissions can also have metadata associated with them.

See about_Admin_Filtering for information about advanced filtering options.

Related topics

Add-AdminPermission
Remove-AdminPermission
Get-AdminRole
Get-AdminPermissionGroup
Test-AdminAccess

Parameters

-Name <String>

Gets permissions with the specified name (localized)

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-Id <String>

Gets permissions with the specified identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>-Description</strong>&lt;String&gt;</td>
<td>Gets permissions with the specified description.</td>
</tr>
<tr>
<td><strong>-GroupId</strong>&lt;String&gt;</td>
<td>Gets permissions that are a member of the specified permission group (by group id).</td>
</tr>
<tr>
<td><strong>-GroupName</strong>&lt;String&gt;</td>
<td>Gets permissions that are a member of the specified permission group (by group name).</td>
</tr>
<tr>
<td><strong>-Metadata</strong>&lt;String&gt;</td>
<td>Gets records with matching metadata entries. The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata &quot;abc:x*&quot; matches records with a metadata entry having a key name of &quot;abc&quot; and a value starting with the letter &quot;x&quot;.</td>
</tr>
<tr>
<td><strong>-Operation</strong>&lt;String&gt;</td>
<td>Gets permissions that contain a specific operation.</td>
</tr>
</tbody>
</table>
### -ReadOnly (Boolean)

Gets permissions with the specified value for the ReadOnly flag.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -Property (String[])

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -ReturnTotalRecordCount (SwitchParameter)

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -MaxRecordCount (Int32)

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -Skip (Int32)

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.
- **SortBy**: `<String>`

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |
| Accept Pipeline Input? | false |

- **Filter**: `<String>`

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **AdminAddress**: `<String>`

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Input Type
None You cannot pipe input into this cmdlet.

Return Values
Citrix.DelegatedAdmin.Sdk.Permission

Get-AdminPermission returns an object for each matching permission.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> Get-AdminPermission -Name *Edit*
Finds all permissions with 'Edit' in their names.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-AdminPermission -Operation "Broker:SetCatalog"
C:\PS> Get-AdminPermission -Filter { Operations -contains "Broker:SetCatalog" -or Operations -contains "Broker:NewCatalog" }
Finds permissions that contain specific operations, with the -Filter form needed to match multiple values.
Get-AdminPermissionGroup

Oct 10, 2013
Gets permission groups configured for the site.

Syntax


Detailed Description

Retrieves permission groups matching the specified criteria. If no parameters are specified this cmdlet enumerates all permission groups.

Permission groups are configured using the Import-AdminRoleConfiguration command, and are primarily used to store the localized name for a group of permissions. Permission groups can also have metadata associated with them.

See about_Admin_Filtering for information about advanced filtering options.

Related topics

Import-AdminRoleConfiguration

Get-AdminPermission

Parameters

-Name <String>
Gets permission groups matching the given name. This is the localized name of the group.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-Id <String>

Gets permission groups with the given identifier. This is the non-localized identifier used to associate permissions with permission groups.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Metadata <String>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1820
Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: `-Metadata "abc:x*"` matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

`-Property [String]`<br>
Specifies the properties to be returned. This is similar to piping the output of the command through `Select-Object`, but the properties are filtered more efficiently at the server.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

`-ReturnTotalRecordCount [SwitchParameter]`<br>
When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

`-MaxRecordCount [Int32]`<br>
Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |
| Accept Pipeline Input? | false |

`-Skip [Int32]`<br>
Skips the specified number of records before returning results. Also reduces the count returned by `-ReturnTotalRecordCount`. 
-SortBy<String>
Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Required? false
Default Value The default sort order is by name or unique identifier.
Accept Pipeline Input? false

-Filter<String>
 Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

Required? false
Default Value
Accept Pipeline Input? false

-AdminAddress<String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

Required? false
Default Value Localhost. Once a value is provided by any cmdlet, this value becomes the default.
Accept Pipeline Input? false

Input Type
None You cannot pipe input into this cmdlet.

Return Values
Citrix.DelegatedAdmin.Sdk.PermissionGroup

Get-AdminPermissionGroup returns an object for each matching permission group.

Examples
-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-AdminPermissionGroup -Name *s
Finds all permission groups that end with the letter 's'.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> $list = @("Hosts","Other")
C:\PS> Get-AdminPermissionGroup -Filter { Id -in $list } -SortBy Name
Retrieve two specific permission group objects with the matching identifiers.
Get-AdminRevision

Oct 10, 2013
Gets the current revision of the delegated administration configuration data.

Syntax

Get-AdminRevision [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Delegated Administration Service maintains a revision number that is incremented whenever its configuration is changed. This command retrieves the current revision number.

Related topics

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

System.Int32

The configuration revision number

Notes

If Int32.MaxValue is reached the value of the revision number will wrap around to Int32.MinValue.

In some cases the value may be incremented by more than one.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-AdminRevision
Retrieve the current revision number.
Get-AdminRole

Oct 10, 2013
Gets roles configured for this site.

Syntax


Detailed Description

Retrieves roles matching the specified criteria. If no parameters are specified, this cmdlet enumerates all roles.

See about_Admin_Filtering for information about advanced filtering options.

Related topics

New-AdminRole
Set-AdminRole
Rename-AdminRole
Remove-AdminRole
Set-AdminRoleMetadata
Remove-AdminRoleMetadata

Parameters

-Name <String>

Gets roles with the specified name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-Id <Guid>

Gets the role with the specified identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
### -**BuiltIn**<Boolean>

Gets roles with the specified value of the `BuiltIn` flag.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -**Description**<String>

Gets roles with the specified description.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -**Metadata**<String>

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: `-Metadata "abc:x*"` matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -**Permission**<String>

Gets roles that contain a specific permission.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -**Property**<String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through `Select-Object`, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

-MaxRecordCount

Specifies the maximum number of records to return.

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.
### -Filter

A string that represents a PowerShell-style filter expression. See about_Admin Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress

The address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

None  
You cannot pipe input into this cmdlet.

**Return Values**

Citrix.DelegatedAdmin.Sdk.Role

Get-AdminRole returns an object for each matching role.

**Examples**

-------------- EXAMPLE 1 --------------

C:\PS> Get-AdminRole -Id 20852cdf-f527-4953-ba6e-e7545217122d

Gets the details of the role with the specific id.

-------------- EXAMPLE 2 --------------

C:\PS> Get-AdminRole -BuiltIn $false

List all custom roles.
Get-AdminRoleConfiguration

Oct 10, 2013

Gets role configurations for this site.

Syntax


Detailed Description

Retrieves role configurations matching the specified criteria. If no parameters are specified, this cmdlet enumerates and returns all role configurations.

Role configurations are part of the product configuration and define what permissions, permission groups, and built-in roles the product has. This cmdlet also provides the mapping of permissions to operations.

Related topics

Import-AdminRoleConfiguration

Parameters

-Name <String>

Gets role configurations matching the specified name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-Id <Guid>

Gets role configurations with the specified id.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-Locale <String>

Gets role configurations with the specified locale. Role configurations usually have a consistent locale.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1829
<table>
<thead>
<tr>
<th>Default Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Priority**<Int32>

 Gets role configurations with the specified priority.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Version**<String>

  Gets role configurations with the matching version number.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Property**<String[]>

  Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ReturnTotalRecordCount**<SwitchParameter>

  When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **MaxRecordCount**<Int32>

  Specifies the maximum number of records to return.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Skip&lt;Int32&gt;</td>
<td>false</td>
<td>250</td>
<td>false</td>
</tr>
<tr>
<td>-SortBy&lt;String&gt;</td>
<td>false</td>
<td>The default sort order is by name or unique identifier.</td>
<td>false</td>
</tr>
<tr>
<td>-Filter&lt;String&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-AdminAddress&lt;String&gt;</td>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

-Skip<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

-Filter<String>

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.
Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.DelegatedAdmin.Sdk.RoleConfiguration

Get-AdminRoleConfiguration returns an object for each matching role configuration.

Notes

This command is supplied for infrastructure purposes only and is not intended for public use.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-AdminRoleConfiguration -Name Director
Retrieve the role configuration for the Citrix Director component.
Get-AdminScope

Oct 10, 2013
Gets scopes configured for this site.

Syntax


Detailed Description

Retrieves scopes matching the specified criteria. If no parameters are specified this cmdlet enumerates all scopes.

There is one special built-in scope, the 'All' scope.

To determine what objects are currently in a scope, use the Get-<Prefix>ScopedObject from each of the relevant PowerShell snap-ins.

See about_Admin_Filtering for information about advanced filtering options.

Related topics

New-AdminScope
Set-AdminScope
Rename-AdminScope
Remove-AdminScope
Set-AdminScopeMetadata
Remove-AdminScopeMetadata

Parameters

-Name <String>

Gets scopes with the specified name.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-Id <Guid>

Gets the scope with the specified identifier.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

- **-BuiltIn**: <Boolean>

Gets scopes with the specified value of the BuiltIn flag.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

- **-Description** : <String>

Gets scopes with the specified description.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

- **-Metadata** : <String>

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*" matches records with a metadata entry having a key name of "abc" and a value starting with the letter "x".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

- **-Property** : <String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>
-ReturnTotalRecordCount &lt;SwitchParameter&gt;

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

-MaxRecordCount &lt;Int32&gt;

Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |
| Accept Pipeline Input? | false |

-Skip &lt;Int32&gt;

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | false |

-SortBy &lt;String&gt;

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |
| Accept Pipeline Input? | false |

-Filter &lt;String&gt;

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.
**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

None You cannot pipe input into this cmdlet.

**Return Values**

Citrix.DelegatedAdmin.Sdk.Scope

Get-AdminScope returns an object for each matching scope.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-AdminScope -Name *Sales*

List all scopes that contain the word 'Sales'.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-AdminScope -Id 21862daf-e529-4553-ba6e-f7543217111e

Gets the details of the scope with the specific id.
Get-AdminService

Oct 10, 2013
Gets the service record entries for the DelegatedAdmin Service.

Syntax


Detailed Description

Returns instances of the DelegatedAdmin Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Related topics

Parameters

-Property <String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ReturnTotalRecordCount <SwitchParameter>

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-MaxRecordCount <Int32>

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>----------------</td>
<td>-----</td>
</tr>
</tbody>
</table>

Accept Pipeline Input?  false

-Skip<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-SortBy<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Filter<String>

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
Citrix.DelegatedAdmin.Sdk.Service

The Get-AdminServiceInstance command returns an object containing the following properties.

Uid <Integer>
Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>
Specifies the unique identifier for the service instance.

DNSName <String>
Specifies the domain name of the host on which the service runs.

MachineName <String>
Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>
Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>
Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>
Specifies the date and time at which the service was last stopped or restarted.

OSType
Specifies the operating system installed on the host on which the service runs.

OSVersion
Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion
Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>
Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>
Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>
Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes
If the command fails, the following errors can be returned.

Error Codes

----------

UnknownObject
One of the specified objects was not found.

PartialData
Only a subset of the available data was returned.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase
The query required to get the database was not defined.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.
Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AdminService

Uid : 1
ServiceHostId : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName : MyServer.company.com
MachineName : MYSERVER
CurrentState : On
LastStartTime : 04/04/2011 15:25:38
LastActivityTime : 04/04/2011 15:33:39
OSType : Win32NT
OSVersion : 6.1.7600.0
ServiceVersion : 5.1.0.0
DatabaseUserName : NT AUTHORITY\NETWORK SERVICE
SID : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {MySiteService1, MySiteService2...}

Get all the instances of the DelegatedAdmin Service running in the current service group.
Get-AdminServiceAddedCapability

Oct 10, 2013

Gets any added capabilities for the DelegatedAdmin Service on the controller.

Syntax

Get-AdminServiceAddedCapability [-AdminAddress <String>] [CommonParameters]

Detailed Description

Enables updates to the DelegatedAdmin Service on the controller to be detected.

You do not need to configure a database connection before using this command.

Related topics

Parameters

- **AdminAddress**:<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>False</td>
</tr>
</tbody>
</table>

Return Values

System.String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

---------

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AdminServiceAddedCapability
Get the added capabilities of the DelegatedAdmin Service.
Get-AdminServiceInstance

Oct 10, 2013
Gets the service instance entries for the DelegatedAdmin Service.

Syntax

Get-AdminServiceInstance [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns service interfaces published by the instance of the DelegatedAdmin Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Related topics

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

Return Values

Citrix.DelegatedAdmin.Sdk.ServiceInstance

The Get-AdminServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.
ServiceType <String>
Specifies the service instance type. For this service, the service instance type is always Admin.

Address
Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding
Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version
Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>
Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>
Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>
Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations
InterService - for operations between different services
Peer - for communications between services of the same type

Metadata <Citrix.DelegatedAdmin.Sdk.Metadata[]>
The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes
If the command fails, the following errors can be returned.

Error Codes

--------

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AdminServiceInstance

Binding            : wcf_HTTP_kerb
InterfaceType      : SDK
Metadata           :
MetadataMap        :
ServiceAccount     : ENG\MyAccount$
ServiceAccountSid  : S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName   : MyServiceGroup
ServiceGroupUid    : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceUid : 00000000-0000-0000-0000-000000000000
ServiceType        : Admin
Version            : 1

Binding            : wcf_HTTP_kerb
InterfaceType      : InterService
Metadata           :
MetadataMap        :
ServiceAccount     : ENG\MyAccount
ServiceAccountSid  : S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName : MyServiceGroup
ServiceGroupUid  : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceId : 00000000-0000-0000-0000-000000000000
ServiceType      : Admin
Version          : 1

Get all instances of the DelegatedAdmin Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.
Get-AdminServiceStatus

Oct 10, 2013

Gets the current status of the DelegatedAdmin Service on the controller.

Syntax

Get-AdminServiceStatus [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Enables the status of the DelegatedAdmin Service on the controller to be monitored. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured.

Related topics

Set-AdminDBConnection
Test-AdminDBConnection
Get-AdminDBConnection
Get-AdminDBSchema

Parameters

-<AdminAddress><String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


The Get-AdminServiceStatus command returns an object containing the status of the DelegatedAdmin Service together with extra diagnostics information.

DBUnconfigured

The DelegatedAdmin Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the DelegatedAdmin Service. This may be because the service attempted
to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the DelegatedAdmin Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBNewerVersionThanService

The version of the DelegatedAdmin Service currently in use is incompatible with the version of the DelegatedAdmin Service schema on the database. Upgrade the DelegatedAdmin Service to a more recent version.

DBOlderVersionThanService

The version of the DelegatedAdmin Service schema on the database is incompatible with the version of the DelegatedAdmin Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The DelegatedAdmin Service is running and is connected to a database containing a valid schema.

Failed

The DelegatedAdmin Service has failed.

Unknown

(0) The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

--------

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AdminServiceStatus

DBUnconfigured
Get the current status of the DelegatedAdmin Service.
Import-AdminRoleConfiguration

Oct 10, 2013
Imports role configuration data into the Delegated Administration Service.

Syntax

[<CommonParameters>]

Import-AdminRoleConfiguration -Content <String> [-Force] [-LoggingId <Guid>] [-AdminAddress <String>]
[<CommonParameters>]

Detailed Description

This command is intended for use by Citrix Studio to import definitions, roles, permissions, and their mappings to operations.

The supplied configuration requires a digital signature; this is used to validate the integrity of the configuration.

Related topics

Get-AdminRoleConfiguration

Parameters

-Path <String>

The path to the file containing the role configuration data.

| Required? | true |
| Default Value | |

Accept Pipeline Input? false

-Content <String>

The content of the role configuration data.

| Required? | true |
| Default Value | |

Accept Pipeline Input? true (ByValue)

-Force <SwitchParameter>

Allows older versions of role configuration to replace newer versions.

| Required? | false |
| Default Value | |

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1851
### LoggingId (Guid)

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### AdminAddress (String)

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

**Input Type**

None You cannot pipe input into this cmdlet.

**Return Values**

None

**Notes**

This command is supplied for infrastructure purposes only and is not intended for public use.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Import-AdminRoleConfiguration -Path 'C:\MyAdminConfig.xml'
Imports the contents of 'C:\MyAdminConfig.xml' to the Delegated Administration Service.
New-AdminAdministrator

Oct 10, 2013
Adds a new administrator to the site.

Syntax

New-AdminAdministrator [-Name] <String> [-Enabled <Boolean>] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

New-AdminAdministrator -Sid <String> [-Enabled <Boolean>] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

New-AdminAdministrator creates a new administrator object in the site. Once a new administrator has been created you can assign rights (role and scope pairs) to the administrator.

Administrator objects are used to determine what rights, and therefore what permissions a particular Active Directory user has through the various SDKs and consoles of the site.

When the Enabled flag of an administrator is set to false, any rights of the administrator are ignored by the system when performing permission checks.

Related topics

Get-AdminAdministrator
Set-AdminAdministrator
Remove-AdminAdministrator
Set-AdminAdministratorMetadata
Remove-AdminAdministratorMetadata

Parameters

-Name <String>

Specifies the user or group name in Active Directory that this administrator corresponds to.

| Required? | True |
| Default Value | |
| Accept Pipeline Input? | True (ByPropertyName) |

-Sid <String>

Specifies the SID (security identifier) of the user in Active Directory that this administrator corresponds to.

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
- `Enabled` <Boolean>

Specifies whether the new administrator starts off enabled or not.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>True</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- `LoggingId` <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- `AdminAddress` <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.DelegatedAdmin.Sdk.Administrator

The newly created administrator.

Examples
-------------------------- EXAMPLE 1 --------------------------

C:\PS> New-AdminAdministrator -Name DOMAIN\TestUser
Creates a new administrator object for user "DOMAIN\TestUser". It defaults to enabled.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> New-AdminAdministrator -Sid S-1-2-34-1234567890-1234567890-1234567890-123
Creates a new administrator object for user with SID "S-1-2-34-1234567890-1234567890-1234567890-123". It defaults to enabled.
New-AdminRole

Oct 10, 2013
Adds a new custom role to the site.

Syntax

New-AdminRole [-Name <String>] [-Description <String>] [-LoggingId <Guid>] [-AdminAddress <String>]
[<CommonParameters>]

Detailed Description

New-AdminRole adds a new custom role object to the site. Once a new role has been created, you can add permissions to the role which define what operations the role conveys.

Roles represent a job function, such as 'help desk administrator', and contain a list of permissions that are required to perform that job function.

To assign a role to an administrator, you combine it with a scope which indicates what objects the role can operate on. This pair (also known as a 'right') can then be assigned to an administrator. See Add-AdminRight for further details.

The identifier of the new role is chosen automatically, and custom roles created with this cmdlet always have their BuiltIn flag set to false.

You cannot modify built-in roles, and only some license editions support custom roles.

Related topics

Get-AdminRole
Set-AdminRole
Rename-AdminRole
Remove-AdminRole
Set-AdminRoleMetadata
Remove-AdminRoleMetadata
Add-AdminPermission
Add-AdminRight

Parameters

-Name <String>

Specifies the name of the role. Each role in a site must have a unique name.
## Accept Pipeline Input?
- **true** (ByPropertyName)

### Description

- **<String>**

Specifies the description of the role.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### LoggingId

- **<Guid>**

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### AdminAddress

- **<String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### Input Type
None

### Return Values

*Citrix.DelegatedAdmin.Sdk.Role*

The newly created role.

### Notes

Roles are created without any permissions. Use the Add-AdminPermission to add permissions.

### Examples
--- EXAMPLE 1 ---

C:\PS> New-AdminRole -Name Supervisor -Description "My custom supervisor role"
C:\PS> $list = Get-AdminRole 'Help Desk Administrator' | Select -Expand Permissions
C:\PS> Add-AdminPermission -Role Supervisor -Permission $list
C:\PS> Add-AdminPermission -Role Supervisor -Permission $extras
C:\PS> Add-AdminRight -Administrator DOMAIN\TestUser -Role Supervisor -All

Creates a new role called 'Supervisor', and then copies the permissions from the help desk role and adds some extras. Then gives this role (with the all scope) to user 'TestUser'.
New-AdminScope

Oct 10, 2013
Adds a new scope to the site.

Syntax

New-AdminScope [-Name] <String> [-Description <String>] [-LoggingId <Guid>] [-AdminAddress <String>]
[<CommonParameters>]

Detailed Description

New-AdminScope adds a new scope object to the site.

A scope represents a collection of objects. Scopes are used to group objects in a way that is relevant to the organization; for example, the set of delivery groups used by the Sales team.

You can create objects in particular scopes by specifying the -Scope parameter of a New- cmdlet for an object that can be scoped. You can then modify the contents of a scope with Add-<Noun>Scope and Remove-<Noun>Scope cmdlets from the corresponding PowerShell snap-ins.

To assign a scope to an administrator, combine it with a role and then assign this pair (also known as a 'right') to an administrator. See Add-AdminRight for further details.

The identifier of the new scope is chosen automatically.

Related topics

Get-AdminScope
Set-AdminScope
Rename-AdminScope
Remove-AdminScope
Set-AdminScopeMetadata
Remove-AdminScopeMetadata
Add-AdminRight

Parameters

-Name <String>

Specifies the name of the scope. Each scope in a site must have a unique name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
- **Description**: <String>

Specifies the description of the scope.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **LoggingId**: <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**: <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

None You cannot pipe input into this cmdlet.

Return Values

Citrix.DelegatedAdmin.Sdk.Scope

The newly created scope.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS> New-AdminScope -Name Sales -Description "Sales department scope"
C:\PS> Add-HypHypervisorConnectionScope -HypervisorConnectionName XenServer2 -Scope Sales
C:\PS> Add-AdminRight -Administrator DOMAIN\TestUser -Role Hosting -Scope Sales
Creates a new scope called 'Sales', adds a hypervisor connection object to the scope, and then assigns the right to use the hosting role on the Sales scope to the 'TestUser' administrator.
Remove-AdminAdministrator

Oct 10, 2013
Removes administrators from the site.

Syntax

Remove-AdminAdministrator [-InputObject] <Administrator[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AdminAdministrator -Sid <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AdminAdministrator [-Name] <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Remove-AdminAdministrator cmdlet deletes administrators from the site.

Related topics

New-AdminAdministrator
Get-AdminAdministrator
Set-AdminAdministrator
Set-AdminAdministratorMetadata
Remove-AdminAdministratorMetadata

Parameters

-**InputObject**<Administrator[]>
  Specifies the administrators to delete.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Name**<String[]>
  Specifies the name of the administrator to delete.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>
-Sid<String[]>

Specifies the SID of the administrator to delete.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-LoggingId<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.DelegatedAdmin.Sdk.Administrator You can pipe the administrators to be deleted into this command.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-AdminAdministrator DOMAIN\TestUser
Remove the administrator called "DOMAIN\TestUser".

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-AdminAdministrator -Enabled $false | Remove-AdminAdministrator
Remove all disabled administrators.
Remove-AdminAdministratorMetadata

Oct 10, 2013
Removes metadata from the given Administrator.

Syntax

Remove-AdminAdministratorMetadata -AdministratorSid <String> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-AdminAdministratorMetadata -AdministratorSid <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-AdminAdministratorMetadata [-AdministratorName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-AdminAdministratorMetadata [-AdministratorName] <String> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-AdminAdministratorMetadata [-InputObject] <Administrator[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-AdminAdministratorMetadata [-InputObject] <Administrator[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Provides the ability to remove metadata from the given Administrator.

Related topics

Set-AdminAdministratorMetadata

Parameters

-AdministratorName <String>

Name of the Administrator

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-InputObject <Administrator[]>

Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
</table>

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1865
<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-AdministratorSid**<String>

Sid of the Administrator

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Name**<String>

The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Map**<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

Error Codes

--------

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AdminAdministrator | % { Remove-AdminAdministratorMetadata -Map $_.MetadataMap }
Remove all metadata from all Administrator objects.
Remove-AdminPermission

Oct 10, 2013
Remove permissions from the set of permissions of a role.

Syntax

Remove-AdminPermission [-InputObject] <Permission[]> -Role <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AdminPermission [-Permission] <String[]> -Role <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AdminPermission -All -Role <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Remove permissions from the set of permissions that a role maps to.

Any administrator with a right including that role immediately loses the ability to use the operations of the removed permissions.

Duplicate permissions do not produce an error, and permissions that the roles does not already have are skipped (without error).

You cannot modify the permissions of built-in roles.

Related topics

Add-AdminPermission
Get-AdminPermission
Get-AdminRole
Get-AdminPermissionGroup
Test-AdminAccess

Parameters

-InputObject <Permission[]>

Specifies the permissions to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Permission <String[]>
Specifies the list of permissions to remove (by identifier).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **Role**:<String>

Role name or identifier of the role to update.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **All**:<SwitchParameter>

Remove all permissions.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**:<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**:<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Input Type

`Citrix.DelegatedAdmin.Sdk.Permission` You can pipe a list of permissions to be removed into this command.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

```powershell
C:\PS> Remove-AdminPermission -Role MyRole -Permission Global_Read,Logging_Read
Remove a couple of specific permissions from the 'MyRole' role.
```

-------------------------- EXAMPLE 2 --------------------------

```powershell
C:\PS> Remove-AdminPermission -Role MyRole -All
Remove all permissions from the 'MyRole' role.
```
Remove-AdminRight

Oct 10, 2013
Removes rights from an administrator.

Syntax

Remove-AdminRight -Scope <String> -Role <String> -Administrator <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AdminRight -Role <String> -Administrator <String> [-All] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AdminRight [-InputObject] <Right[]> -Administrator <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

This command removes rights from the specified administrator.

For convenience, you can use the -All parameter to specify the 'All' scope.

Use the Get-AdminAdministrator cmdlet to determine what rights an administrator has.

Related topics

Get-AdminAdministrator
Get-AdminEffectiveRight
Add-AdminRight

Parameters

-InputObject <Right[]>

Specifies the rights to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Scope <String>

Specifies the scope name or scope identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
-Role <String>
Specifies the role name or role identifier.

  Required?  true
  Default Value
  Accept Pipeline Input?  false

-Administrator <String>
Specifies the name or SID of the administrator.

  Required?  true
  Default Value
  Accept Pipeline Input?  false

-All <SwitchParameter>
Specifies the 'All' scope. This parameter avoids localization issues or having to type the identifier of the 'All' scope.

  Required?  false
  Default Value
  Accept Pipeline Input?  false

-LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

  Required?  false
  Default Value
  Accept Pipeline Input?  false

-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.
RemoveAdminRight

Required? false
Default Value Localhost. Once a value is provided by any cmdlet, this value becomes the default.
Accept Pipeline Input? false

Input Type
Citrix.DelegatedAdmin.Sdk.Right You can pipe the rights to be removed into this command.

Return Values
None

Examples
--------------- EXAMPLE 1 --------------------------
C:\PS> RemoveAdminRight -Role 'Help Desk Administrator' -Scope London -Administrator DOMAIN\Admin1
Removes the 'Help Desk Administrator' role and 'London' scope from user 'Admin1'

--------------- EXAMPLE 2 --------------------------
C:\PS> $admin = Get-AdminAdministrator -Name DOMAIN\Admin
C:\PS> Remove-AdminRight -InputObject $admin.Rights -Administrator DOMAIN\Admin
Removes all rights from administrator 'Admin'.
Remove-AdminRole

Oct 10, 2013
Removes a role from the site.

Syntax

Remove-AdminRole [-InputObject] <Role[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]
Remove-AdminRole [-Id] <Guid[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]
Remove-AdminRole [-Name] <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Remove-AdminRole cmdlet deletes roles from the site.

You cannot remove built-in roles or roles that are currently assigned to an administrator.

Related topics

New-AdminRole
Get-AdminRole
Set-AdminRole
Rename-AdminRole
Set-AdminRoleMetadata
Remove-AdminRoleMetadata

Parameters

-InputObject <Role[]>
Specifies the roles to remove (by role object).

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Id <Guid[]>
Specifies the roles to remove (by role id).

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | true (ByPropertyName)
--- | ---

**-Name**<String[]>

Specifies the roles to remove (by role name).

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Input Type

Citrix.DelegatedAdmin.Sdk.Role You can pipe the roles to be deleted into this command.

Return Values

None

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Remove-AdminRole -Name Supervisor
Remove the Supervisor role.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> Get-AdminRole -BuiltIn $false | Remove-AdminRole
Attempt to remove all custom roles. This fails if one of the roles is assigned to an administrator.
Remove-AdminRoleMetadata

Oct 10, 2013
Removes metadata from the given Role.

Syntax

Remove-AdminRoleMetadata [-RoleId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AdminRoleMetadata [-RoleId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AdminRoleMetadata [-RoleName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AdminRoleMetadata [-RoleName] <String> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AdminRoleMetadata [-InputObject] <Role[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AdminRoleMetadata [-InputObject] <Role[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given Role.

Related topics

Set-AdminRoleMetadata

Parameters

-**RoleId**<Guid>
  
  Id of the Role

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-**RoleName**<String>
  
  Name of the Role

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
</table>

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1878
### Default Value

| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

### -InputObject <Role[]>

Objects to which the metadata is to be added.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

### -Map <PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with `@{“name1” = “val1”; “name2” = “val2”}`) or a string dictionary (created with `new-object “System.Collections.Generic.Dictionary[String,String]”`). The properties whose names match keys in the map will be removed.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

### -Name <String>

The metadata property to remove.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

### -LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### -AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>InvalidParameterCombination</td>
</tr>
<tr>
<td>The cmdlet parameters are inconsistent.</td>
</tr>
<tr>
<td>UnknownObject</td>
</tr>
<tr>
<td>One of the specified objects was not found.</td>
</tr>
<tr>
<td>DatabaseError</td>
</tr>
<tr>
<td>An error occurred in the service while attempting a database operation.</td>
</tr>
<tr>
<td>DatabaseNotConfigured</td>
</tr>
<tr>
<td>The operation could not be completed because the database for the service is not configured.</td>
</tr>
<tr>
<td>DataStoreException</td>
</tr>
<tr>
<td>An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.</td>
</tr>
<tr>
<td>PermissionDenied</td>
</tr>
<tr>
<td>You do not have permission to execute this command.</td>
</tr>
<tr>
<td>AuthorizationError</td>
</tr>
<tr>
<td>There was a problem communicating with the Citrix Delegated Administration Service.</td>
</tr>
<tr>
<td>ConfigurationLoggingError</td>
</tr>
<tr>
<td>The operation could not be performed because of a configuration logging error.</td>
</tr>
<tr>
<td>CommunicationError</td>
</tr>
<tr>
<td>There was a problem communicating with the remote service.</td>
</tr>
<tr>
<td>ExceptionThrown</td>
</tr>
</tbody>
</table>
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AdminRole | % { Remove-AdminRoleMetadata -Map $_.MetadataMap }

Remove all metadata from all Role objects.
Remove-AdminScope

Oct 10, 2013
Removes a scope from the site.

Syntax

Remove-AdminScope [-InputObject] <Scope[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AdminScope [-Id] <Guid[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AdminScope [-Name] <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Remove-AdminScope cmdlet deletes scopes from the site.

You cannot remove the built-in 'All' scope or scopes that are currently assigned to an administrator.

Related topics

New-AdminScope
Get-AdminScope
Set-AdminScope
Rename-AdminScope
Set-AdminScopeMetadata
Remove-AdminScopeMetadata

Parameters

-**InputObject** <Scope[]>

Specifies the scopes to remove (by scope object).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Id** <Guid[]>

Specifies the scopes to remove (by scope id).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | true (ByPropertyName)
---|---

**-Name**<String[]>

Specifies the scopes to remove (by scope name).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type  
Citrix.DelegatedAdmin.Sdk.Scope You can pipe the scopes to be deleted into this command.

Return Values
None

Examples

------------- EXAMPLE 1 -------------
C:\PS> Remove-AdminScope -Name Sales
Remove the Sales scope.

------------- EXAMPLE 2 -------------
C:\PS> Get-AdminScope -BuiltIn $false | Remove-AdminScope
Attempt to remove all scopes (excluding the built-in 'All' scope). This fails if one of the scopes is assigned to an administrator.
Remove-AdminScopeMetadata

Oct 10, 2013
Removes metadata from the given Scope.

Syntax

Remove-AdminScopeMetadata [-Scopeld] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AdminScopeMetadata [-Scopeld] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AdminScopeMetadata [-ScopeName] <String> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AdminScopeMetadata [-ScopeName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AdminScopeMetadata [-InputObject] <Scope[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-AdminScopeMetadata [-InputObject] <Scope[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given Scope.

Related topics

Set-AdminScopeMetadata

Parameters

-**Scopeld**<Guid>

Id of the Scope

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-**ScopeName**<String>

Name of the Scope

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
</table>
### Default Value

| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

### InputObject <Scope[]>

Objects to which the metadata is to be added.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

### Map <PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with `new-object "System.Collections.Generic.Dictionary[String,String]"`). The properties whose names match keys in the map will be removed.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

### Name <String>

The metadata property to remove.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

### LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

### AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AdminScope | % { Remove-AdminScopeMetadata -Map $_.MetadataMap }
Remove all metadata from all Scope objects.
Remove-AdminServiceMetadata

Oct 10, 2013
Removes metadata from the given Service.

Syntax

Remove-AdminServiceMetadata [-ServiceHostId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AdminServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AdminServiceMetadata [-InputObject] <Service[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-AdminServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given Service.

Related topics

Set-AdminServiceMetadata

Parameters

-**ServiceHostId**<Guid>

Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-**InputObject**<Service[]>

Objects to which metadata is to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Name**<String>
The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Map**<PSObject>

Specifies a dictionary of {name, value}-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

**Error Codes**

---------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-AdminService | % { Remove-AdminServiceMetadata -Map $_.MetadataMap }
Remove all metadata from all Service objects.
Rename-AdminRole

Oct 10, 2013
Rename a role

Syntax

Rename-AdminRole [-InputObject] <Role> [-NewName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Rename-AdminRole [-Id] <Guid> [-NewName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Rename-AdminRole [-Name] <String> [-NewName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Rename-AdminRole cmdlet changes the name of a role.

Role names must be unique, and you cannot modify the name of built-in roles.

Related topics

New-AdminRole
Get-AdminRole
Set-AdminRole
Remove-AdminRole
Set-AdminRoleMetadata
Remove-AdminRoleMetadata

Parameters

- **InputObject** <Role>

Specifies the role to rename (by object).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Id** <Guid>

Specifies the role to rename (by id).
### -Name <String>

Specifies the role to rename (by name).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### -NewName <String>

Specifies the new name of the role.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -PassThru <SwitchParameter>

Returns the affected record. By default, this cmdlet does not generate any output.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.DelegatedAdmin.Sdk.Role You can pipe the role to be renamed into this command.

Return Values

None or Citrix.DelegatedAdmin.Sdk.Role

When you use the PassThru parameter, Rename-AdminRole generates a Citrix.DelegatedAdmin.Sdk.Role object. Otherwise, this cmdlet does not generate any output.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-AdminRole -Name Supervisor -NewName HelpDeskLead
Renames the 'Supervisor' role to 'HelpDeskLead'.
Rename-AdminScope

Oct 10, 2013
Rename a scope

Syntax

Rename-AdminScope [-InputObject] <Scope> [-NewName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Rename-AdminScope [-Id] <Guid> [-NewName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Rename-AdminScope [-Name] <String> [-NewName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Rename-AdminScope cmdlet changes the name of a scope.

Scope names must be unique, and you cannot modify the name of the built-in 'All' scope.

Related topics

New-AdminScope

Get-AdminScope

Set-AdminScope

Remove-AdminScope

Set-AdminScopeMetadata

Remove-AdminScopeMetadata

Parameters

-InputObject <Scope>

Specifies the scope to rename (by object).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Id <Guid>

Specifies the scope to rename (by id).
<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-Name**<String>

Specifies the scope to rename (by name).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

**-NewName**<String>

Specifies the new name of the scope.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-PassThru**<SwitchParameter>

Returns the affected record. By default, this cmdlet does not generate any output.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.DelegatedAdmin.Sdk.Scope You can pipe the scope to be renamed into this command.

**Return Values**

None or Citrix.DelegatedAdmin.Sdk.Scope

When you use the PassThru parameter, Rename-AdminScope generates a Citrix.DelegatedAdmin.Sdk.Scope object. Otherwise, this cmdlet does not generate any output.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Rename-AdminScope -Name Sales -NewName SalesDesktops

Renames the 'Sales' scope to 'SalesDesktops'.

--------------------------
Reset-AdminServiceGroupMembership

Oct 10, 2013
Rebads the access permissions and configuration service locations for the DelegatedAdmin Service.

Syntax

Reset-AdminServiceGroupMembership [-ConfigServiceInstance] <ServiceInstance[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Enables you to reload DelegatedAdmin Service access permissions and configuration service locations. The Reset-AdminServiceGroupMembership command must be run on at least one instance of the service type (Admin) after installation and registration with the configuration service. Without this operation, the DelegatedAdmin services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-AdminServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Related topics

Parameters

- **-ConfigServiceInstance**: Specifies the configuration service instance object that represents the service instance for the type 'InterService' that references a configuration service for the deployment.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue) |

- **-LoggingId**: Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

- **-AdminAddress**: Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |
Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-AdminServiceGroupMembership command.

Notes
If the command fails, the following errors can be returned.

Error Codes
----------

NoSuitableServiceInstance
None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-AdminServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config -AdminAddress OtherServer.example.com | Reset-AdminServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.
Set-AdminAdministrator

Oct 10, 2013
Sets the properties of an administrator.

Syntax


Set-AdminAdministrator -Sid <String[]> [-Enabled <Boolean>] [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminAdministrator [-Name] <String[]> [-Enabled <Boolean>] [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Set-AdminAdministrator cmdlet is used to enable or disable an existing administrator.

You can specify the administrators to modify in a number of ways, by piping in existing objects, by passing existing objects with the InputObject parameter, or by specifying the names or SIDs explicitly.

Related topics

New-AdminAdministrator
Get-AdminAdministrator
Remove-AdminAdministrator
Set-AdminAdministratorMetadata
Remove-AdminAdministratorMetadata

Parameters

-InputObject <Administrator[]>

Specifies the administrators to modify.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-Name <String[]>

Specifies the names of the administrators to modify.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1900
### Sid <String[]>
Specifies the SIDs of the administrators to modify.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

### Enabled <Boolean>
Specifies the new value for the Enabled property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### PassThru <SwitchParameter>
Returns the affected record. By default, this cmdlet does not generate any output.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name.
or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.DelegatedAdmin.Sdk.Administrator You can pipe the administrators to be modified into this command.

**Return Values**

None or Citrix.DelegatedAdmin.Sdk.Administrator

When you use the PassThru parameter, Set-AdminAdministrator generates a Citrix.DelegatedAdmin.Sdk.Administrator object. Otherwise, this cmdlet does not generate any output.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Get-AdminAdministrator -Enabled $false | Set-AdminAdministrator -Enabled $true
Enable all administrators that are currently disabled.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Set-AdminAdministrator -Name DOMAIN\TestUser1,DOMAIN\TestUser2 -Enabled $true
Enable two specific users specified by name (TestUser1 and TestUser2).
Set-AdminAdministratorMetadata

Oct 10, 2013
Adds or updates metadata on the given Administrator.

Syntax

Set-AdminAdministratorMetadata -AdministratorSid <String> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminAdministratorMetadata -AdministratorSid <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminAdministratorMetadata [-AdministratorName] <String> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminAdministratorMetadata [-AdministratorName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminAdministratorMetadata [-InputObject] <Administrator[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminAdministratorMetadata [-InputObject] <Administrator[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability for additional custom data to be stored against given Administrator objects.

Related topics

Remove-AdminAdministratorMetadata

Parameters

-AdministratorName <String>
Name of the Administrator

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-InputObject <Administrator[]>
Objects to which the metadata is to be added.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-AdministratorSid <String>
Sid of the Administrator
| **Required?** | true |
| **Default Value** | |
| **Accept Pipeline Input?** | true (ByPropertyName) |

**-Name**: `<String>`

Specifies the property name of the metadata to be added. The property must be unique for the Administrator specified. The property cannot contain any of the following characters "/:#.*?=<>[]()".

| **Required?** | true |
| **Default Value** | |
| **Accept Pipeline Input?** | false |

**-Value**: `<String>`

Specifies the value for the property.

| **Required?** | true |
| **Default Value** | |
| **Accept Pipeline Input?** | false |

**-Map**: `<PSObject>`

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with `new-object "System.Collections.Generic.Dictionary[String,String]"`).

| **Required?** | true |
| **Default Value** | |
| **Accept Pipeline Input?** | true (ByValue) |

**-LoggingId**: `<Guid>`

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| **Required?** | false |
| **Default Value** | |
| **Accept Pipeline Input?** | false |

**-AdminAddress**: `<String>`

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| **Required?** | false |
| **Default Value** | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
Accept Pipeline Input? | false

Return Values


Set-AdminAdministratorMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>
Specifies the name of the property.

Value <string>
Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.
### EXAMPLE 1

```powershell
c:\PS> Set-AdminAdministratorMetadata -AdministratorSid S-1-5-21-1505241163-3345470479-1241728991-1000 -Name property -Value value
```

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of `property` and a value of `value` to the Administrator with the identifier `S-1-5-21-1505241163-3345470479-1241728991-1000`. 
Set-AdminDBConnection

Oct 10, 2013
Configures a database connection for the DelegatedAdmin Service.

Syntax

Set-AdminDBConnection [-DBConnection] <String> [-Force] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Configures a connection to a database in which the DelegatedAdmin Service can store its state. The service will attempt to connect and start using the database immediately after the connection is configured. The database connection string is updated to the specified value regardless of whether it is valid or not. Specifying an invalid connection string prevents a service from functioning until the error is corrected.

After a connection is configured, you cannot alter it without first clearing it (by setting the connection to $null).

You do not need to configure a database connection to use this command.

Related topics

Get-AdminServiceStatus
Get-AdminDBConnection
Test-AdminDBConnection

Parameters

- **-DBConnection** <String>

  Specifies the database connection string to be used by the DelegatedAdmin Service. Passing in $null will clear any existing database connection configured.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Force** <SwitchParameter>

  If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
- **AdminAddress**: String

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>localhost</td>
<td>false</td>
</tr>
</tbody>
</table>

  *Default Value*: Localhost. Once a value is provided by any cmdlet, this value becomes the default.

### Return Values

- **Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo**

  The Set-AdminDBConnection command returns an object containing the status of the DelegatedAdmin Service together with extra diagnostics information.

- **DBUnconfigured**
  
  The DelegatedAdmin Service does not have a database connection configured.

- **DBRejectedConnection**
  
  The database rejected the logon attempt from the DelegatedAdmin Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- **InvalidDBConfigured**
  
  The expected stored procedures are missing from the database. This may be because the DelegatedAdmin Service schema has not been added to the database.

- **DBNotFound**
  
  The specified database could not be located with the configured connection string.

- **DBNewerVersionThanService**
  
  The version of the DelegatedAdmin Service currently in use is incompatible with the version of the DelegatedAdmin Service schema on the database. Upgrade the DelegatedAdmin Service to a more recent version.

- **DBOlderVersionThanService**
  
  The version of the DelegatedAdmin Service schema on the database is incompatible with the version of the DelegatedAdmin Service currently in use. Upgrade the database schema to a more recent version.

- **DBVersionChangeInProgress**
  
  A database schema upgrade is currently in progress.

- **OK**
  
  The DelegatedAdmin Service is running and is connected to a database containing a valid schema.

- **Failed**
  
  The DelegatedAdmin Service has failed.

- **Unknown**
  
  The status of the DelegatedAdmin Service cannot be determined.
Notes
If the command fails, the following errors can be returned.

Error Codes
---------

InvalidDBConnectionString
The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured
There was already a database connection configured. After a configuration is set, it can only be set to $null.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Set-AdminDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Configures a database connection string for the DelegatedAdmin Service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Set-AdminDBConnection -DBConnection "Invalid Connection String"

Invalid database connection string format.
Configures an invalid database connection string for the DelegatedAdmin Service.
Set-AdminRole

Oct 10, 2013
Set the properties of a role.

Syntax


Detailed Description

The Set-AdminRole command allows the description of custom roles to be updated. You cannot modify built-in roles.

To modify the permissions of a role, use the Add-AdminPermission and Remove-AdminPermission cmdlets.

To update the metadata associated with a role, use the Set-AdminRoleMetadata and Remove-AdminRoleMetadata cmdlets.

Related topics

New-AdminRole
Get-AdminRole
Remove-AdminRole
Rename-AdminRole
Set-AdminRoleMetadata
Remove-AdminRoleMetadata
Add-AdminPermission
Remove-AdminPermission

Parameters

-InputObject <Role[]>

Specifies the roles to update (by object).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputObject</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
<td></td>
</tr>
<tr>
<td><strong>-Id</strong>&lt;Guid[]&gt;</td>
<td>Specifies the roles to update (by id).</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
<td></td>
</tr>
<tr>
<td><strong>-Name</strong>&lt;String[]&gt;</td>
<td>Specifies the roles to update (by name).</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
<td></td>
</tr>
<tr>
<td><strong>-Description</strong>&lt;String&gt;</td>
<td>Supplies the new description value.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td><strong>-PassThru</strong>&lt;SwitchParameter&gt;</td>
<td>Returns the affected record. By default, this cmdlet does not generate any output.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>False</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td><strong>-LoggingId</strong>&lt;Guid&gt;</td>
<td>Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.</td>
<td></td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.DelegatedAdmin.Sdk.Role You can pipe the roles to be updated into this command.

**Return Values**

None or Citrix.DelegatedAdmin.Sdk.Role

When you use the PassThru parameter, Set-AdminRole generates a Citrix.DelegatedAdmin.Sdk.Role object. Otherwise, this cmdlet does not generate any output.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-AdminRole -Name Supervisor -Description "Helpdesk supervisor role"
Change the description of the ‘Supervisor’ role.
Set-AdminRoleMetadata

Oct 10, 2013
Adds or updates metadata on the given Role.

Syntax

Set-AdminRoleMetadata [-RoleId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters>

Set-AdminRoleMetadata [-RoleId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters>

Set-AdminRoleMetadata [-RoleName] <String> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters>

Set-AdminRoleMetadata [-RoleName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters>

Set-AdminRoleMetadata [-InputObject] <Role[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters>

Set-AdminRoleMetadata [-InputObject] <Role[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters>

Detailed Description

Provides the ability for additional custom data to be stored against given Role objects.

Related topics

Remove-AdminRoleMetadata

Parameters

-**RoleId**<Guid>

Id of the Role

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-**RoleName**<String>

Name of the Role

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Accept Pipeline Input?</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **InputObject**<Role[]>

Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Map**<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name**<String>

Specifies the property name of the metadata to be added. The property must be unique for the Role specified. The property cannot contain any of the following characters \/#:.*?=<>|[]()"'

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Value**<String>

Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create
high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**


Set-AdminRoleMetadata returns a dictionary containing the new (name, value)-pairs.

**Key <string>**

Specifies the name of the property.

**Value <string>**

Specifies the value for the property.

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

-----------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.
DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> Set-AdminRoleMetadata -RoleId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of ‘property’ and a value of ‘value’ to the Role with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.
Set-AdminScope

Oct 10, 2013
Set the properties of a scope.

Syntax

Set-AdminScope [-InputObject] <Scope[]> [-Description <String>] [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Set-AdminScope [-Id] <Guid[]> [-Description <String>] [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Set-AdminScope [-Name] <String[]> [-Description <String>] [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The Set-AdminScope command allows the description of scopes to be updated. You cannot modify the built-in 'All' scope.

To change the contents of a scope, use the Add-<Noun>Scope and Remove-<Noun>Scope cmdlets from the corresponding PowerShell snap-in.

To update the metadata associated with a scope, use the Set-AdminScopeMetadata and Remove-AdminScopeMetadata cmdlets.

Related topics

New-AdminScope
Get-AdminScope
Remove-AdminScope
Rename-AdminScope
Set-AdminScopeMetadata
Remove-AdminScopeMetadata

Parameters

- **-InputObject** <Scope[]>

  Specifies the scopes to update (by object).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
## -Id

Specifies the scopes to update (by id).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

## -Name

Specifies the scopes to update (by name).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

## -Description

Supplies the new description value.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

## -PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

## -LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
### -AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

Citrix.DelegatedAdmin.Sdk.Scope You can pipe the scopes to be updated into this command.

**Return Values**

None or Citrix.DelegatedAdmin.Sdk.Scope

When you use the PassThru parameter, Set-AdminScope generates a Citrix.DelegatedAdmin.Sdk.Scope object. Otherwise, this cmdlet does not generate any output.

**Examples**

```
EXAMPLE 1
```

C:\PS> Set-AdminScope -Name Sales -Description "Sales department desktops"
Change the description of the 'Sales' scope.
Set-AdminScopeMetadata

Oct 10, 2013
Adds or updates metadata on the given Scope.

Syntax

Set-AdminScopeMetadata [-ScopeId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminScopeMetadata [-ScopeId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminScopeMetadata [-ScopeName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminScopeMetadata [-ScopeName] <String> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminScopeMetadata [-InputObject] <Scope[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminScopeMetadata [-InputObject] <Scope[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability for additional custom data to be stored against given Scope objects.

Related topics

Remove-AdminScopeMetadata

Parameters

- **ScopeId** <Guid>

Id of the Scope

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **ScopeName** <String>

Name of the Scope

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.1920
-**Input Object** `<Scope[]>`

Objects to which the metadata is to be added.

| Required? | true |
| Default Value | true (ByValue) |
| Accept Pipeline Input? | false |

-**Name** `<String>`

Specifies the property name of the metadata to be added. The property must be unique for the Scope specified. The property cannot contain any of the following characters: `/;:.*?<>|[]()"'

| Required? | true |
| Default Value | true (ByValue) |
| Accept Pipeline Input? | false |

-**Value** `<String>`

Specifies the value for the property.

| Required? | true |
| Default Value | true |
| Accept Pipeline Input? | false |

-**Map** `<PSObject>`

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with `new-object "System.Collections.Generic.Dictionary[String,String]"`).

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-**LoggingId** `<Guid>`

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create
high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress** <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


Set-AdminScopeMetadata returns a dictionary containing the new (name, value)-pairs.

**Key** <string>

Specifies the name of the property.

**Value** <string>

Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

**Error Codes**

------------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.
DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS> Set-AdminScopeMetadata -Scopeld 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Scope with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Set-AdminServiceMetadata

Oct 10, 2013
Adds or updates metadata on the given Service.

Syntax

Set-AdminServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminServiceMetadata [-ServiceHostId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-AdminServiceMetadata [-InputObject] <Service[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Allows you to store additional custom data against given Service objects.

Related topics
Remove-AdminServiceMetadata

Parameters

-ServiceHostId <Guid>
Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-InputObject <Service[]>
Objects to which metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Map <PSObject>
Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1";

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1924
"name2" = "val2") or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String"]).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-Name <String>**

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/:#.*?=<> | ]]["

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Value <String>**

Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId <Guid>**

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress <String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values


Set-AdminServiceMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>
Specifies the name of the property.

Value <string>
Specifies the value for the property.

Notes
If the command fails, the following errors can be returned.

Error Codes
----------

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Set-AdminServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2A06DD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DD0805C'.
Test-AdminAccess

Oct 10, 2013
Retrieves the scopes where the specified operation is permitted.

Syntax

Test-AdminAccess [-Operation] <String> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

This cmdlet evaluates what rights the current user has, and from these determines the scopes where the specified operation is permitted.

Operations are the indivisible unit of functionality that each XenDesktop service can perform, and usually correspond to individual cmdlets.

Related topics

Parameters

- **-Operation**<String>

  The operation to query.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-AdminAddress**<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.DelegatedAdmin.Sdk.ScopeReference

The list of permissible scopes for the specified operation.

Notes

If the specified operation has unrestricted access a single object is returned representing the 'All' scope with a ScopeId of
Guid.Empty (00000000-0000-0000-0000-000000000000).

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Test-AdminAccess -Operation 'Broker:GetCatalog'
Queries the scopes where 'Broker:GetCatalog' is permitted.
Test-AdminDBConnection

Oct 10, 2013
Tests a database connection for the DelegatedAdmin Service.

Syntax

Test-AdminDBConnection [-DBConnection] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Tests a connection to the database in which the DelegatedAdmin Service can store its state. The service will attempt to connect to the database without affecting the current connection to the database.

You do not have to clear the connection to use this command.

Related topics

Get-AdminServiceStatus
Get-AdminDBConnection
Set-AdminDBConnection

Parameters

- **DBConnection**<String>

Specifies the database connection string to be tested by the DelegatedAdmin Service.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

- **LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values
The Test-AdminDBConnection command returns an object containing the status of the DelegatedAdmin Service if the connection string of the specified data store were to be set to the string being tested, together with extra diagnostics information for the specified connection string.

**DBRejectedConnection**

The database rejected the logon attempt from the DelegatedAdmin Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**

The expected stored procedures are missing from the database. This may be because the DelegatedAdmin Service schema has not been added to the database.

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**

The version of the DelegatedAdmin Service currently in use is incompatible with the version of the DelegatedAdmin Service schema on the database. Upgrade the DelegatedAdmin Service to a more recent version.

**DBOlderVersionThanService**

The version of the DelegatedAdmin Service schema on the database is incompatible with the version of the DelegatedAdmin Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**

A database schema upgrade is currently in progress.

**OK**

The Set-AdminDBConnection command would succeed if it were executed with the supplied connection string.

**Failed**

The DelegatedAdmin Service has failed.

**Unknown**

The status of the DelegatedAdmin Service cannot be determined.

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

---------

**InvalidDBConnectionString**

The database connection string has an invalid format.

**PermissionDenied**

You do not have permission to execute this command.

**AuthorizationError**

There was a problem communicating with the Citrix Delegated Administration Service.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Test-AdminDBConnection -DBConnection "Server=servername\SQLEXPRESS;Initial Catalog = dbName; Integrated Security = True"
OK
Tests a database connection string for the DelegatedAdmin Service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Test-AdminDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Tests an invalid database connection string for the DelegatedAdmin Service.
Overview

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnvTestEnvTestSnapin</td>
<td>The Citrix Environment Test Service provides tools to test and inspect the state of a XenDesktop installation.</td>
</tr>
<tr>
<td>EnvTest Filtering</td>
<td>Describes the common filtering options for XenDesktop cmdlets.</td>
</tr>
</tbody>
</table>

Cmdlets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get-EnvTestConfiguration</td>
<td>Gets the Environment Test Service’s configuration options</td>
</tr>
<tr>
<td>Get-EnvTestDBConnection</td>
<td>Gets the database string for the specified data store used by the EnvTest Service.</td>
</tr>
<tr>
<td>Get-EnvTestDBSchema</td>
<td>Gets a script that creates the EnvTest Service database schema for the specified data store.</td>
</tr>
<tr>
<td>Get-EnvTestDBVersionChangeScript</td>
<td>Gets a script that updates the EnvTest Service database schema.</td>
</tr>
<tr>
<td>Get-EnvTestDefinition</td>
<td>Gets the one or more test definitions</td>
</tr>
<tr>
<td>Get-EnvTestInstalledDBVersion</td>
<td>Gets a list of all available database schema versions for the EnvTest Service.</td>
</tr>
<tr>
<td>Get-EnvTestService</td>
<td>Gets the service record entries for the EnvTest Service.</td>
</tr>
<tr>
<td>Get-EnvTestServiceAddedCapability</td>
<td>Gets any added capabilities for the EnvTest Service on the controller.</td>
</tr>
<tr>
<td>Get-EnvTestServiceInstance</td>
<td>Gets the service instance entries for the EnvTest Service.</td>
</tr>
<tr>
<td>Get-EnvTestServiceStatus</td>
<td>Gets the current status of the EnvTest Service on the controller.</td>
</tr>
<tr>
<td>Get-EnvTestSuiteDefinition</td>
<td>Gets one or more test suite definitions</td>
</tr>
<tr>
<td>Get-EnvTestTask</td>
<td>Gets one or more EnvTestTask(s)</td>
</tr>
<tr>
<td>New-EnvTestDiscoveryTargetDefinition</td>
<td>Creates a new EnvTestDiscoveryTargetDefinition object</td>
</tr>
<tr>
<td>Remove-EnvTestServiceMetadata</td>
<td>Removes metadata from the given Service.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Remove-EnvTestTask</td>
<td>Removes from the database completed tasks for the EnvTest Service.</td>
</tr>
<tr>
<td>Remove-EnvTestTaskMetadata</td>
<td>Removes metadata from the given Task.</td>
</tr>
<tr>
<td>Reset-EnvTestServiceGroupMembership</td>
<td>Reloads the access permissions and configuration service locations for the EnvTest Service.</td>
</tr>
<tr>
<td>Set-EnvTestConfiguration</td>
<td>Sets the Environment Test Service's configuration options</td>
</tr>
<tr>
<td>Set-EnvTestDBConnection</td>
<td>Configures a database connection for the EnvTest Service.</td>
</tr>
<tr>
<td>Set-EnvTestServiceMetadata</td>
<td>Adds or updates metadata on the given Service.</td>
</tr>
<tr>
<td>Set-EnvTestTaskMetadata</td>
<td>Adds or updates metadata on the given Task.</td>
</tr>
<tr>
<td>Start-EnvTestTask</td>
<td>Starts a new test task.</td>
</tr>
<tr>
<td>Stop-EnvTestTask</td>
<td>Stops a still running task from completing.</td>
</tr>
<tr>
<td>Switch-EnvTestTask</td>
<td>Sets the current task that will be returned by a call to Get-EnvTestTask with no parameters.</td>
</tr>
<tr>
<td>Test-EnvTestDBConnection</td>
<td>Tests a database connection for the EnvTest Service.</td>
</tr>
</tbody>
</table>
TOPIC

about_EnvTestEnvTestSnapin

SHORT DESCRIPTION

The Citrix Environment Test Service provides tools to test and inspect the state of a XenDesktop installation.

COMMAND PREFIX

All commands in this snap-in have the noun prefixed with 'EnvTest'.

LONG DESCRIPTION

The Citrix Environment Test Service provides tools to test and inspect the state of a XenDesktop installation at different points during and after configuration and install.
about_EnvTest_Filtering

Oct 10, 2013

TOPIC

XenDesktop - Advanced Dataset Filtering

SHORT DESCRIPTION

Describes the common filtering options for XenDesktop cmdlets.

LONG DESCRIPTION

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

```
Get-<Noun> | Where { $_.Size = 'Small' } | Sort 'Date' | Select -First 10
```

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

```
Get-<Noun> -Size 'Small' -SortBy 'Date' -MaxRecordCount 10
```

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

- `MaxRecordCount <int>`

  Specifies the maximum number of results to return.
  For example, to return only the first nine results use:

  ```
  Get-<Noun> -MaxRecordCount 9
  ```

  If not specified, only the first 250 records are returned, and if more are available, a warning is produced:
WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

To retrieve all records, specify a large number for -MaxRecordCount. As the value is an integer, you can use the following:

```
Get-<Noun> -MaxRecordCount [int]:MaxValue
```

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
```

Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: () [Get-<Noun>], PartialDataException
+ FullyQualifiedName     : PartialData,Citrix.<SDKName>.SDK.Get<Noun>

The count can be accessed using the TotalAvailableResultCount property:

```
$Count = $error[0].TotalAvailableResultCount
```

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.
Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

```
-SortBy 'Name,-Count'
```

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values. Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

```
-SortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'
```

```
-Filter <String>
```

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

```
Get-
```
See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
Get-<Noun> -Company "Abc[*]"                 # Matches Abc*
Get-<Noun> -Company "Abc`*"                  # Matches Abc*
Get-<Noun> -Filter { Company -eq "Abc`*" }  # Matches Abc*
Get-<Noun> -Filter { Company -eq "A`B`C" }   # Matches A`B`C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

```
Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```
Get-<Noun> -Filter 'Enabled'      # Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled' # Equivalent to 'Enabled -eq $false'
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```
Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:
By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

```
Get-<Noun> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z" }
$d = [DateTime]"2010-08-23T12:30:00.0Z"
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }
```

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

```
Get-<Noun> -Filter { StartTime -ge '-2' }         # Two days ago
Get-<Noun> -Filter { StartTime -ge '-1:30' }      # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge '-0:0:30' }    # 30 seconds ago
```

ARRAY PROPERTIES
When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings. Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

```
Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*" }
Get-<Noun> -Filter { Users -contains "Fred*" }
```

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter { User -eq "Frederick" }
# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter { User -lt 'F' }
```

COMPLEX EXPRESSIONS
When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

```powershell
Get-<Noun> -Filter { Shape -eq 'Circle' -or Shape -eq 'Square' }
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }
```

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

```powershell
Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue' -and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue' -and Shape -eq 'Circle') }
```

**PAGING**

The simplest way to page through data is to use the -Skip and -MaxRecordCount parameters. So, to read the first three pages of data with 10 records per page, use:

```powershell
Get-<Noun> -Skip 0 -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 10 -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 20 -MaxRecordCount 10 <other filtering criteria>
```

You must include the same filtering criteria on each call, and ensure that the data is sorted consistently.

The above approach is often acceptable, but as each call performs an independent query, data changes can result in records being skipped or appearing twice. One approach to improve this is to sort by a unique id field and then start the search for the next page at the unique id after the last unique id of the previous page. For example:

```powershell
# Get the first page
Get-<Noun> -MaxRecordCount 10 -SortBy SerialNumber

SerialNumber ... 
----------- ...
A120004
A120007
... 7 other records ...
A120900

# Get the next page
Get-<Noun> -MaxRecordCount 10 -Filter { FirstName -gt 'A120900' }

SerialNumber ... 
```
FILTER SYNTAX DEFINITION

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= "{" <ComponentList> "}" 

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> | 

  <Component>

<Component> ::= <NotOperator> <Factor> | 

  <Factor>

<Factor> ::= "(" <ComponentList> ")" | 

  <PropertyName> <ComparisonOperator> <Value> | 
  <PropertyName>

<AndOrOperator> ::= "-and" | "-or" 

<NotOperator> ::= "-not" | "!"

<ComparisonOperator>

  ::= "-eq" | "-ne" | "-le" | "-ge" | "-lt" | "-gt" |
  "-like" | "-notlike" | "-contains" | "-notcontains" |
  "-in" | "-notin"

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> | 

  <scalar variable> | <array variable> | 
  "$null" | "$true" | "$false"
Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30" means 1 hour and 30 minutes ago.
Get-EnvTestConfiguration

Oct 10, 2013
Gets the Environment Test Service's configuration options

Syntax
Get-EnvTestConfiguration [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Gets the Environment Test Service's configuration options and returns them as key/value pairs.

Related topics

Parameters

- **LoggingId**<Guid>
  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>
  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Return Values

Dictionary<string, object>
All configuration settings

Examples

------------------------ EXAMPLE 1 ------------------------
Get-EnvTestConfiguration
Gets all configuration options
Get-EnvTestDBConnection

Oct 10, 2013
Gets the database string for the specified data store used by the EnvTest Service.

Syntax

Get-EnvTestDBConnection [-AdminAddress <String>] [CommonParameters>

Detailed Description

Returns the database connection string for the specified data store.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

Related topics

Get-EnvTestServiceStatus
Set-EnvTestDBConnection
Test-EnvTestDBConnection

Parameters

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Return Values

system: string

The database connection string configured for the EnvTest Service.

Notes

If the command fails, the following errors can be returned.

Error Codes

---------

NoDBConnections
The database connection string for the EnvTest Service has not been specified.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-EnvTestDBConnection

Server=servername\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True
Get the database connection string for the EnvTest Service.
Get-EnvTestDBSchema

Oct 10, 2013

Gets a script that creates the EnvTest Service database schema for the specified data store.

Syntax

Get-EnvTestDBSchema [-DatabaseName <String>] [-ServiceGroupName <String>] [-ScriptType <ScriptTypes>] [-LocalDatabase] [-Sid <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Gets SQL scripts that can be used to create a new EnvTest Service database schema, add a new EnvTest Service to an existing site, remove an EnvTest Service from a site, or create a database server logon for an EnvTest Service. If no Sid parameter is provided, the scripts obtained relate to the currently selected EnvTest Service instance, otherwise the scripts relate to EnvTest Service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied. The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to an EnvTest SDK cmdlet. The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured. The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SMDCMD mode'. The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to EnvTest Service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to EnvTest Service roles

If ScriptType is Evict, the returned script contains:

- Removal of EnvTest Service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

If the service uses two data stores they can exist in the same database. You do not need to configure a database before using this command.
Related topics

Set-EnvTestDBConnection

Parameters

**-DatabaseName**<String>

Specifies the name of the database for which the schema will be generated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ServiceGroupName**<String>

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the EnvTest services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ScriptType**<ScriptTypes>

Specifies the type of database script returned. Available script types are:

**Database**

Returns a full database script that can be used to create a database schema for the EnvTest Service in a database instance that does not already contain a schema for this service. The DatabaseName and ServiceGroupName parameters must be specified to create a script of this type.

**Instance**

Returns a permissions script that can be used to add further EnvTest services to an existing database instance that already contains the full EnvTest service schema, associating the services to the Service Group. The Sid parameter can optionally be specified to create a script of this type.

**Login**

Returns a database logon script that can be used to add the required logon accounts to an existing database instance that contains the EnvTest Service schema. This is used primarily when creating a mirrored database environment. The DatabaseName parameter must be specified to create a script of this type.

**Evict**
Returns a script that can be used to remove the specified EnvTest Service from the database entirely. The DatabaseName and Sid parameters must be specified to create a script of this type.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Database</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LocalDatabase** `<SwitchParameter>`

  Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for EnvTest services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Sid** `<String>`

  Specifies the SID of the controller on which the EnvTest Service instance to remove from the database is running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-AdminAddress** `<String>`

  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.String

A string containing the required SQL script for application to a database.

Notes
The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

Create the database schema.
Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned.

Error Codes
-----------

GetSchemasFailed
The database schema could not be found.
ActiveDirectoryAccountResolutionFailed
The specified Active Directory account or Group could not be found.
DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.
PermissionDenied
You do not have permission to execute this command.
AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-EnvTestDBSchema -DatabaseName MyDB -ServiceGroupName MyServiceGroup > c:\EnvTestSchema.sql
Get the full database schema for site data store of the EnvTest Service and copy it to a file called 'c:\EnvTestSchema.sql'.

This script can then be used to create the schema in a pre-existing database named 'MyDB' that does not already contain a EnvTest Service site schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-EnvTestDBSchema -DatabaseName MyDB -scriptType Login > c:\EnvTestLogins.sql
Get the logon scripts for the EnvTest Service.
Get-EnvTestDBVersionChangeScript

Oct 10, 2013
Gets a script that updates the EnvTest Service database schema.

Syntax
Get-EnvTestDBVersionChangeScript -DatabaseName <String> -TargetVersion <Version> [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description
Returns a database script that can be used to upgrade or downgrade the site or secondary schema for the EnvTest Service from the current schema version to a different version.

Related topics
Get-EnvTestInstalledDBVersion

Parameters
-DatabaseName <String>
Specifies the name of the database instance to which the update applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-TargetVersion <Version>
Specifies the version of the database you want to update to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values

System.Management.Automation.PSObject

A PSObject containing the required SQL script for application to a database.

Notes

The PSObject returned by this cmdlet contains the following properties:

-- Script The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

-- NeedExclusiveAccess Indicates whether all services in the service group must be shut down during the update or not.

-- CanUndo Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any EnvTest services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-EnvTestServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned.

Error Codes

-----------

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the EnvTest Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $update = Get-EnvTestDBVersionChangeScript -DatabaseName MyDb -TargetVersion 1.0.75.0
C:\PS> $update.Script > update_75.sql

 Gets an SQL update script to update the current schema to version 1.0.75.0. The resulting update_75.sql script is suitable for direct use with the SQL Server SQLCMD utility.
Get-EnvTestDefinition

Oct 10, 2013
Gets the one or more test definitions

Syntax

Get-EnvTestDefinition [-TestId <String[]>] [-CultureName <String>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Returns a list of test definitions that are available from currently running components.

Related topics

Get-EnvTestSuiteDefinition
Get-EnvTestTask
Start-EnvTestTask
Switch-EnvTestTask
Stop-EnvTestTask
Remove-EnvTestTask
Add-EnvTestTaskMetadata
Remove-EnvTestTaskMetadata

Parameters

-TestId <String[]>
The id of one or more tests.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

-CultureName <String>
The culture name in which to produce results. The culture name is in standard language/region-code format; for example "en-US".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
- **AdminAddress**: String

  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**


**Return Values**

Citrix.EnvTest.Sdk.EnvTestDefinition

One or more test definitions.

**Examples**

------------ EXAMPLE 1 --------------

```powershell
$allTestDefinitions = Get-EnvTestDefinition
Retrieve all tests.
```

------------ EXAMPLE 2 --------------

```powershell
$allTestDefinitionsTranslatedIntoSpanish = Get-EnvTestDefinition -CultureName es-ES
Retrieve all tests with localized properties returned in Spanish.
```

------------ EXAMPLE 3 --------------

```powershell
$monitorConfigServiceRegistrationDefinition = Get-EnvTestDefinition -TestId Monitor_RegisteredWithConfigurationService
Retrieve the definition of the 'Monitor_RegisteredWithConfigurationService' test.
```
Get-EnvTestInstalledDBVersion

Oct 10, 2013

Gets a list of all available database schema versions for the EnvTest Service.

Syntax

Get-EnvTestInstalledDBVersion [-Upgrade] [-Downgrade] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Returns the current version of the EnvTest Service database schema, if no flags are set, otherwise returns versions for which upgrade or downgrade scripts are available and have been stored in the database.

Related topics

Parameters

- **-Upgrade**<SwitchParameter>

Specifies that only schema versions to which the current database version can be updated should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Downgrade**<SwitchParameter>

Specifies that only schema versions to which the current database version can be reverted should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
System.Version

The Get-EnvTestInstalledDbVersion command returns objects containing the new definition of the EnvTest Service database schema version.

Major <Integer>
Minor <Integer>
Build <Integer>
Revision <Integer>

Notes
If the command fails, the following errors can be returned.

Error Codes
-----------

InvalidParameterCombination
Both the Upgrade and Downgrade flags were specified.

NoOp
The operation was successful but had no effect.

NoDBConnections
The database connection string for the EnvTest Service has not been specified.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-EnvTestInstalledDBVersion

<table>
<thead>
<tr>
<th>Major</th>
<th>Minor</th>
<th>Build</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Get the currently installed version of the EnvTest Service database schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-EnvTestInstalledDBVersion -Upgrade

<table>
<thead>
<tr>
<th>Major</th>
<th>Minor</th>
<th>Build</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Get the versions of the EnvTest Service database schema for which upgrade scripts are supplied.
Get-EnvTestService

Oct 10, 2013
Gets the service record entries for the EnvTest Service.

Syntax


Detailed Description

Returns instances of the EnvTest Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Related topics

Parameters

-Property <String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ReturnTotalRecordCount <SwitchParameter>

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_EnvTest_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-MaxRecordCount <Int32>

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
**Default Value**

<table>
<thead>
<tr>
<th>Default Value</th>
<th>250</th>
</tr>
</thead>
</table>

**Accept Pipeline Input?**

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>false</th>
</tr>
</thead>
</table>

**-Skip<Int32>**

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SortBy<String>**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter<String>**

Gets records that match a PowerShell-style filter expression. See about_EnvTest_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**
The Get-EnvTestServiceInstance command returns an object containing the following properties.

Uid <Integer>
Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>
Specifies the unique identifier for the service instance.

DNSName <String>
Specifies the domain name of the host on which the service runs.

MachineName <String>
Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>
Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>
Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>
Specifies the date and time at which the service was last stopped or restarted.

OSType
Specifies the operating system installed on the host on which the service runs.

OSVersion
Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion
Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>
Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>
Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>
Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

---------

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.
Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-EnvTestService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
SID                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {MySiteService1, MySiteService2...}

Get all the instances of the EnvTest Service running in the current service group.
Get-EnvTestServiceAddedCapability

Oct 10, 2013
Gets any added capabilities for the EnvTest Service on the controller.

Syntax
Get-EnvTestServiceAddedCapability [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Enables updates to the EnvTest Service on the controller to be detected.
You do not need to configure a database connection before using this command.

Related topics
Parameters

- **-AdminAddress**<String>
  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

Return Values
System.String
String containing added capabilities.

Notes
If the command fails, the following errors can be returned.

Error Codes

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-EnvTestServiceAddedCapability
Get the added capabilities of the EnvTest Service.
Get-EnvTestServiceInstance

Oct 10, 2013
Gets the service instance entries for the EnvTest Service.

Syntax

Get-EnvTestServiceInstance [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns service interfaces published by the instance of the EnvTest Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Related topics

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | false

Return Values

Citrix.EnvTest.Sdk.ServiceInstance

The Get-EnvTestServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.
ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always EnvTest.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.EnvTest.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

--------

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-EnvTestServiceInstance

Address : http://MyServer.com:80/Citrix/EnvTestContract
Binding : wcf_HTTP_kerb
InterfaceType : SDK
Metadata :
MetadataMap :
ServiceAccount : ENG\MyAccount$
ServiceAccountSid : S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName : MyServiceGroup
ServiceGroupUid : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceUid : 00000000-0000-0000-0000-000000000000
ServiceType : EnvTest
Version : 1

Address : http://MyServer.com:80/Citrix/EnvTestContract/IServiceApi
Binding : wcf_HTTP_kerb
InterfaceType : InterService
Metadata :
MetadataMap :
ServiceAccount : ENG\MyAccount
ServiceAccountSid : S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName : MyServiceGroup
ServiceGroupUid : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceUid : 00000000-0000-0000-0000-000000000000
ServiceType : EnvTest
Version : 1

Get all instances of the EnvTest Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.
Get-EnvTestServiceStatus

Oct 10, 2013
Gets the current status of the EnvTest Service on the controller.

Syntax

Get-EnvTestServiceStatus [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Enables the status of the EnvTest Service on the controller to be monitored. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured.

Related topics

Set-EnvTestDBConnection
Test-EnvTestDBConnection
Get-EnvTestDBConnection
Get-EnvTestDBSchema

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


The Get-EnvTestServiceStatus command returns an object containing the status of the EnvTest Service together with extra diagnostics information.

DBUnconfigured

The EnvTest Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the EnvTest Service. This may be because the service attempted to log on
with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfiguration

The expected stored procedures are missing from the database. This may be because the EnvTest Service schema has not been added to the database.

DBNotFoundException

The specified database could not be located with the configured connection string.

DBNewerVersionThanService

The version of the EnvTest Service currently in use is incompatible with the version of the EnvTest Service schema on the database. Upgrade the EnvTest Service to a more recent version.

DBOlderVersionThanService

The version of the EnvTest Service schema on the database is incompatible with the version of the EnvTest Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The EnvTest Service is running and is connected to a database containing a valid schema.

Failed

The EnvTest Service has failed.

Unknown

(0) The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-EnvTestServiceStatus

DBUnconfigured
Get the current status of the EnvTest Service.
Get-EnvTestSuiteDefinition

Oct 10, 2013
Gets one or more test suite definitions.

Syntax
Get-EnvTestSuiteDefinition [-TestSuiteId <String[]>] [-CultureName <String>] [-AdminAddress <String>]
[<CommonParameters>]

Detailed Description
Returns a list of test suite definitions that are available from currently running components.

Related topics
Get-EnvTestDefinition
Get-EnvTestTask
Start-EnvTestTask
Switch-EnvTestTask
Stop-EnvTestTask
Remove-EnvTestTask
Add-EnvTestTaskMetadata
Remove-EnvTestTaskMetadata

Parameters
-TestSuiteId <String[]>
The id of one or more test suites.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

-CultureName <String>
The culture name in which to produce results. The culture name is in standard language/region-code format; for example "en-US".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.1975
-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type


Return Values

Citrix.EnvTest.Sdk.EnvTestSuiteDefinition

The definition of a test suite

Examples

-------------------------- EXAMPLE 1 --------------------------

$allTestSuiteDefinitions = Get-EnvTestSuiteDefinition
Retrieve all test suites.

-------------------------- EXAMPLE 2 --------------------------

$allTestSuiteDefinitionsTranslatedIntoSpanish = Get-EnvTestSuiteDefinition -CultureName es-ES
Retrieve all test suites with localized properties returned in Spanish.

-------------------------- EXAMPLE 3 --------------------------

$infrastructureSuiteDefinition = Get-EnvTestSuiteDefinition -TestSuiteId Infrastructure
Retrieve the definition of the 'Infrastructure' test suite.
Get-EnvTestTask

Oct 10, 2013
Gets one or more EnvTestTask(s)

Syntax

Get-EnvTestTask [-TaskId <Guid>] [-List] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Returns either the current task, a specified task, or list of tasks that are currently known to the EnvTest Service.

Related topics

Get-EnvTestDefinition
Get-EnvTestSuiteDefinition
Start-EnvTestTask
Switch-EnvTestTask
Stop-EnvTestTask
Remove-EnvTestTask
Add-EnvTestTaskMetadata
Remove-EnvTestTaskMetadata

Parameters

-TaskId<Guid>

Specifies the task identifier to be returned. This value can be retrieved from an existing task's $task.TaskId property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-List<SwitchParameter>

List all running tasks, including the current one.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**:<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

  **Return Values**

  Citrix.EnvTest.Sdk.EnvTestTask

  A description of a previously started task.

  **Examples**

  --------------- EXAMPLE 1 ---------------

  $currentTask = Get-EnvTestTask

  Retrieve the current task. The current task is the most recently created task unless Switch-EnvTestTask explicitly changes it.

  --------------- EXAMPLE 2 ---------------

  $taskOfSpecificId = Get-EnvTestTask -TaskId 36C0EC52-2039-4D6E-B690-9F02F8CEFFCC

  Retrieve a fresh copy of a task object based on a known task id, which is always a Guid. This id can be retrieved from an existing task object via its $Task.TaskId property.

  --------------- EXAMPLE 3 ---------------

  $allKnownTasks = Get-EnvTestTask -List

  Retrieve the list of current tasks. This list includes any task started by the Start-EnvTestTask cmdlet since the service started that has not later been removed via Remove-EnvTestTask.
New-EnvTestDiscoveryTargetDefinition

Oct 10, 2013
Creates a new EnvTestDiscoveryTargetDefinition object

Syntax

New-EnvTestDiscoveryTargetDefinition -TestId <String> [-TargetIdType <String>] [-TargetId <String>] [-AdminAddress <String>] [<CommonParameters>]
New-EnvTestDiscoveryTargetDefinition -TestSuiteId <String> [-TargetIdType <String>] [-TargetId <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Creates a new EnvTestDiscoveryTargetDefinition object that can be piped into Start-EnvTestTask to define one or more targets of execution, optionally including root objects for discovery.

Related topics

Get-EnvTestDefinition
Get-EnvTestSuiteDefinition
Get-EnvTestTask
Start-EnvTestTask
Switch-EnvTestTask
Stop-EnvTestTask
Remove-EnvTestTask
Add-EnvTestTaskMetadata
Remove-EnvTestTaskMetadata

Parameters

-TestId <String>
Test identifiers. If specified, do not specify -TestSuiteId.

| Required? | true |
| Default Value | Empty |
| Accept Pipeline Input? | false |

-TestSuiteId <String>
Test suite identifiers. If specified, do not specify -TestId.

| Required? | true |
| Default Value | Empty |
| Accept Pipeline Input? | false |

-TargetIdType <String>
Describes the type of corresponding object passed with -TargetId
- **TestId**<String>

The Ids that object tests or test suites will target. By default, other components are queried for objects related to these.

| Required? | false |
| Default Value | Empty |
| Accept Pipeline Input? | false |

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values

Citrix.EnvTest.Sdk.EnvTestDiscoveryTargetDefinition

Defines a target of a task

Examples

-------------------------- EXAMPLE 1 --------------------------

```powershell
$singleSimpleTestTaskTarget = New-EnvTestDiscoveryTargetDefinition -TestId Monitor_RegisteredWithConfigurationService
$singleSimpleTestTaskTarget | Start-EnvTestTask
```

Create a discovery target definition with a single test and no target object, then start a task based on it.

-------------------------- EXAMPLE 2 --------------------------

```powershell
$singleSimpleTestSuiteTaskTarget = New-EnvTestDiscoveryTargetDefinition -TestSuiteId Infrastructure
$singleSimpleTestSuiteTaskTarget | Start-EnvTestTask
```

Create a discovery target definition with a single test suite and no target object, then start a task based on it.

-------------------------- EXAMPLE 3 --------------------------

```powershell
$singleTestSuiteTaskTarget = New-EnvTestDiscoveryTargetDefinition -TestSuiteId Catalog -TargetIdType Catalog -TargetId $(Get-BrokerCatalog).Uuid
$singleTestSuiteTaskTarget | Start-EnvTestTask
```

Create a discovery target definition with a single test suite and a catalog target object, then start a task based on it.

-------------------------- EXAMPLE 4 --------------------------

```powershell
$singleSimpleTestSuiteTaskTarget = New-EnvTestDiscoveryTargetDefinition -TestSuiteId Infrastructure
$singleTestSuiteTaskTarget = New-EnvTestDiscoveryTargetDefinition -TestSuiteId Catalog -TargetIdType Catalog -TargetId $(Get-BrokerCatalog).Uuid
@($singleSimpleTestSuiteTaskTarget, $singleTestSuiteTaskTarget) | Start-EnvTestTask
```

Create two different discovery target definitions, put them in an array, then start a task based on both.
Remove-EnvTestServiceMetadata

Oct 10, 2013
Removes metadata from the given Service.

Syntax

Remove-EnvTestServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-EnvTestServiceMetadata [-ServiceHostId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-EnvTestServiceMetadata [-InputObject] <Service[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-EnvTestServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Provides the ability to remove metadata from the given Service.

Related topics

Set-EnvTestServiceMetadata

Parameters

- **ServiceHostId**<Guid>

  Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **InputObject**<Service[]>

  Objects to which metadata is to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Map**<PSObject>
Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

**-Name**<String>

The metadata property to remove.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

---------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------- EXAMPLE 1 ---------------
c:\PS>Get-EnvTestService | % { Remove-EnvTestServiceMetadata -Map $_.MetadataMap }  
Remove all metadata from all Service objects.
Remove-EnvTestTask

Oct 10, 2013
Removes from the database completed tasks for the EnvTest Service.

Syntax

Remove-EnvTestTask [-TaskId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-EnvTestTask [-Task <EnvTestTask>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Enables completed tasks that have run within the EnvTest Service to be removed from the database.

Related topics

Get-EnvTestDefinition
Get-EnvTestSuiteDefinition
Get-EnvTestTask
Start-EnvTestTask
Switch-EnvTestTask
Stop-EnvTestTask
Add-EnvTestTaskMetadata
Remove-EnvTestTaskMetadata

Parameters

-TaskId <Guid>

Specifies the identifier of the task to be removed, retrieveable from the $task.TaskId property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Task <EnvTestTask>

Specifies the task to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | true (ByValue)

- AdminAddress:<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Examples

-------------------------- EXAMPLE 1 --------------------------

Remove-EnvTestTask
Removes the current task from the EnvTest service.

-------------------------- EXAMPLE 2 --------------------------

Remove-EnvTestTask -TaskId 50A4139F-2B55-4A97-A1BE-20EE4E124AA3
Removes a task from the EnvTest service via its id, which is available from an existing task's $task.TaskId property.

-------------------------- EXAMPLE 3 --------------------------

$secondTask = $(Get-EnvTestTask -List)[1]
Remove-EnvTestTask -Task $secondTask
Removes the second task in the list returned by Get-EnvTestTask -List.
Remove-EnvTestTaskMetadata

Oct 10, 2013
Removes metadata from the given Task.

Syntax

Remove-EnvTestTaskMetadata [-TaskTaskId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-EnvTestTaskMetadata [-TaskTaskId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-EnvTestTaskMetadata [-InputObject] <EnvTestTask[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-EnvTestTaskMetadata [-InputObject] <EnvTestTask[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Provides the ability to remove metadata from the given Task.

Related topics
Set-EnvTestTaskMetadata

Parameters

-TaskId <Guid>
Id of the Task

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-InputObject <EnvTestTask[]>
Objects to which the metadata is to be added.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-Map <PSObject>
Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | true (ByValue)

- **-Name**<String>

The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | false

- **-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | false

- **-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | false

Notes

If the command fails, the following errors can be returned.

Error Codes

--------------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-EnvTestTask | % { Remove-EnvTestTaskMetadata -Map $_.MetadataMap }
Remove all metadata from all Task objects.
Reset-EnvTestServiceGroupMembership

Oct 10, 2013
Reloads the access permissions and configuration service locations for the EnvTest Service.

Syntax
Reset-EnvTestServiceGroupMembership [-ConfigServiceInstance] <ServiceInstance[]> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]}

Detailed Description
Enables you to reload EnvTest Service access permissions and configuration service locations. The Reset-EnvTestServiceGroupMembership command must be run on at least one instance of the service type (EnvTest) after installation and registration with the configuration service. Without this operation, the EnvTest services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-EnvTestServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Related topics
Parameters
-ConfigServiceInstance<ServiceInstance[]>
Specifies the configuration service instance object that represents the service instance for the type 'InterService' that references a configuration service for the deployment.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-LoggingId<Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-AdminAddress<String>
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Input Type
Citrix.EnvTest.Sdk.ServiceInstance[] Service instances containing a ServiceInstance object that refers to the central configuration service interservice
interface can be piped to the `Reset-EnvTestServiceGroupMembership` command.

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

-------------

**NoSuitableServiceInstance**

None of the supplied service instance objects were suitable for resetting service group membership.

**DatabaseError**

An error occurred in the service while attempting a database operation.

**DatabaseNotConfigured**

The operation could not be completed because the database for the service is not configured.

**DataStoreException**

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

**PermissionDenied**

You do not have permission to execute this command.

**AuthorizationError**

There was a problem communicating with the Citrix Delegated Administration Service.

**ConfigurationLoggingError**

The operation could not be performed because of a configuration logging error.

**CommunicationError**

There was a problem communicating with the remote service.

**ExceptionThrown**

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-EnvTestServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config -AdminAddress OtherServer.example.com | Reset-EnvTestServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.
Set-EnvTestConfiguration

Sets the Environment Test Service's configuration options

Syntax

Set-EnvTestConfiguration [-PerTypeDiscoveredObjectLimit <Int32>] [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

Sets the Environment Test Service's configuration options and broadcasts the changes to other machines in the Site.

Related topics

Parameters

- **-PerTypeDiscoveredObjectLimit**<Int32>

Sits the maximum number of objects of a particular type to be explored when discovering objects during a test run. Default: 50

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |
Accept Pipeline Input? | false

Input Type
Integer Must be greater than zero.

Return Values
System.String

Examples
-------------------------- EXAMPLE 1 --------------------------
Set-EnvTestConfiguration -PerTypeDiscoveredObjectLimit 100
Set the maximum to 100
Set-EnvTestDBConnection

Oct 10, 2013
Configures a database connection for the EnvTest Service.

Syntax

Set-EnvTestDBConnection [-DBConnection] <String> [-Force] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Configures a connection to a database in which the EnvTest Service can store its state. The service will attempt to connect and start using the database immediately after the connection is configured. The database connection string is updated to the specified value regardless of whether it is valid or not. Specifying an invalid connection string prevents a service from functioning until the error is corrected.

After a connection is configured, you cannot alter it without first clearing it (by setting the connection to $null).

You do not need to configure a database connection to use this command.

Related topics

Get-EnvTestServiceStatus
Get-EnvTestDBConnection
Test-EnvTestDBConnection

Parameters

-DBConnection <String>

Specifies the database connection string to be used by the EnvTest Service. Passing in $null will clear any existing database connection configured.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-Force <SwitchParameter>

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

| Required? | false |
| Default Value | false |
| Accept Pipeline Input? | false |

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |
-AdminAddress<"String"

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


The Set-EnvTestDBConnection command returns an object containing the status of the EnvTest Service together with extra diagnostics information.

DBUnconfigured

The EnvTest Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the EnvTest Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the EnvTest Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBNewerVersionThanService

The version of the EnvTest Service currently in use is incompatible with the version of the EnvTest Service schema on the database. Upgrade the EnvTest Service to a more recent version.

DBOlderVersionThanService

The version of the EnvTest Service schema on the database is incompatible with the version of the EnvTest Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The EnvTest Service is running and is connected to a database containing a valid schema.

Failed

The EnvTest Service has failed.

Unknown

The status of the EnvTest Service cannot be determined.

Notes

If the command fails, the following errors can be returned.
Error Codes

------------

InvalidDBConnectionString
The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured
There was already a database connection configured. After a configuration is set, it can only be set to $null.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

----------------------------- EXAMPLE 1 -----------------------------
c:\PS> Set-EnvTestDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"

OK
Confirms a database connection string for the EnvTest Service.

----------------------------- EXAMPLE 2 -----------------------------
c:\PS> Set-EnvTestDBConnection -DBConnection "Invalid Connection String"

Invalid database connection string format.
Confirms an invalid database connection string for the EnvTest Service.
Set-EnvTestServiceMetadata

Oct 10, 2013
Adds or updates metadata on the given Service.

Syntax

Set-EnvTestServiceMetadata [-ServiceHostId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-EnvTestServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-EnvTestServiceMetadata [-InputObject] <Service[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-EnvTestServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Allows you to store additional custom data against given Service objects.

Related topics

Remove-EnvTestServiceMetadata

Parameters

-ServiceHostId <Guid>
  Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-InputObject <Service[]>
  Objects to which metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Name <String>
  Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/:.*=?<>|[]()"
<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Value&lt;String&gt;</td>
<td>true</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Specifies the value for the property.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Map&lt;PSObject&gt;</td>
<td>true</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{&quot;name1&quot; = &quot;val1&quot;; &quot;name2&quot; = &quot;val2&quot;}) or a string dictionary (created with new-object &quot;System.Collections.Generic.Dictionary[String,String]&quot;).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-LoggingId&lt;Guid&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-AdminAddress&lt;String&gt;</td>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
<tr>
<td>Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Set-EnvTestServiceMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>
Specifies the name of the property.

Value <string>
Specifies the value for the property.

Notes
If the command fails, the following errors can be returned.

Error Codes
-----------

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> Set-EnvTestServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Set-EnvTestTaskMetadata

Oct 10, 2013
Adds or updates metadata on the given Task.

Syntax

Set-EnvTestTaskMetadata [-TaskTaskId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-EnvTestTaskMetadata [-TaskTaskId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-EnvTestTaskMetadata [-InputObject] <EnvTestTask[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-EnvTestTaskMetadata [-InputObject] <EnvTestTask[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Provides the ability for additional custom data to be stored against given Task objects.

Related topics
Remove-EnvTestTaskMetadata

Parameters

-TaskTaskId <Guid>
Id of the Task

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-InputObject <EnvTestTask[]>
Objects to which the metadata is to be added.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-Name <String>
Specifies the property name of the metadata to be added. The property must be unique for the Task specified. The property cannot contain any of the following characters \/;*:?=<> |[]"'

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Value**: <String>

Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Map**: <PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**LoggingId**: <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**: <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>
Default Value

<table>
<thead>
<tr>
<th>Default Value</th>
<th>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</th>
</tr>
</thead>
</table>

Accept Pipeline Input?

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>false</th>
</tr>
</thead>
</table>

Return Values


Set-EnvTestTaskMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>

Specifies the name of the property.

Value <string>

Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

-----------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------- EXAMPLE 1 ---------------

c:\PS> Set-EnvTestTaskMetadata -TaskTaskId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Task with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Start-EnvTestTask

Starts a new test task.

Syntax

Start-EnvTestTask -TestId <String> [-TargetIdType <String>] [-TargetId <String>] [-CultureName <String>] [-IgnoreRelatedObjects] [-RunAsynchronously] [-ExcludeNotRunTests] [-AdminAddress <String>] [CommonParameters]

Start-EnvTestTask -TestSuiteId <String> [-TargetIdType <String>] [-TargetId <String>] [-CultureName <String>] [-IgnoreRelatedObjects] [-RunAsynchronously] [-ExcludeNotRunTests] [-AdminAddress <String>] [CommonParameters]

Start-EnvTestTask -InputObject <PSObject[]> [-CultureName <String>] [-IgnoreRelatedObjects] [-RunAsynchronously] [-ExcludeNotRunTests] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Starts a new test task based on a set of criteria provided via parameters or piped input and either waits for the tests to run or returns immediately depending on how it is called. When running a test suite and providing a target object for that suite, the service will discover related objects by default, but this behavior may be disabled if desired.

Related topics

Get-EnvTestDefinition
Get-EnvTestSuiteDefinition
Get-EnvTestTask
New-EnvTestDiscoveryTargetDefinition
Switch-EnvTestTask
Stop-EnvTestTask
Remove-EnvTestTask
Add-EnvTestTaskMetadata
Remove-EnvTestTaskMetadata

Parameters

-TestId <String>
  Test identifiers. If specified, do not specify -TestSuiteId.

  Required? true
  Default Value Empty
  Accept Pipeline Input? false

-TestSuiteId <String>
  Test suite identifiers. If specified, do not specify -TestId.

  Required? true
  Default Value Empty
  Accept Pipeline Input? false

-InputObject <PSObject[]>
  One or more test targets defining the task, see Input Types for details about what kind of objects are permissible. Any valid object passed to this parameter may also be piped into this cmdlet.

  Required? true
  Default Value Empty
  Accept Pipeline Input? true [ByValue]

-TargetIdType <String>
  Describes the type of corresponding object passed with -TargetId

  Required? false
  Default Value Empty
  Accept Pipeline Input? false

-TargetId <String>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
The Ids that object tests or tests suites will target. By default, other components are queried for objects related to these.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Empty</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<CultureName><String>
The culture name in which to produce results. The culture name is in standard language/region-code format; for example "en-US".

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The current user interface culture</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<IgnoreRelatedObjects><SwitchParameter>
Do not ask other components for objects related to a specified target.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<RunAsynchronously><SwitchParameter>
Do not wait for the test run to complete, return immediately.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<ExcludeNotRunTests><SwitchParameter>
If set, tests that are not run because no object matching their requirements is found are NOT included in test counts and results.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False (include Not Run tests)</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

<AdminAddress><String>
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.EnvTest.Sdk.EnvTestDiscoveryTargetDefinition A single EnvTestDiscoveryTargetDefinition can be specified to target one test or test suite. Citrix.EnvTest.Sdk.EnvTestDiscoveryTargetDefinition[] An array of EnvTestDiscoveryTargetDefinition(s) can be specified to target any combination of tests and/or test suites. PSCustomObject A single PSCustomObject with properties matching the required EnvTestDiscoveryTargetDefinition properties can be specified to target one test or test suite. PSCustomObject[] An array of PSCustomObject(s) with properties matching the required EnvTestDiscoveryTargetDefinition properties can be specified to target any combination of tests and/or test suites.

Return Values

Citrix.EnvTest.Sdk.EnvTestTask

The newly started task.

Examples

-------------------------- EXAMPLE 1 --------------------------

$singleSimpleTestTask = Start-EnvTestTask -TestId Monitor_RegisteredWithConfigurationService
Create and start a task with a single test and no target object.

-------------------------- EXAMPLE 2 --------------------------

$singleSimpleTestTaskInSpanish = Start-EnvTestTask -TestId Monitor_RegisteredWithConfigurationService -CultureName es ES
Create and start a task with a single test and no target object, with localized properties translated into Spanish.
----- EXAMPLE 3 -----

```
$singleSimpleTestSuiteTask = Start-EnvTestTask -TestSuiteId Infrastructure
```

Create and start a task with a single test suite and no target object.

----- EXAMPLE 4 -----

```
$singleTestSuiteTask = Start-EnvTestTask -TestSuiteId Catalog -TargetIdType Catalog -TargetId $(Get-BrokerCatalog).Uuid
```

Create and start a task with a single test suite and a catalog target object.

----- EXAMPLE 5 -----

```
$singleTestSuiteTask = Start-EnvTestTask -TestSuiteId Catalog -TargetIdType Catalog -TargetId $(Get-BrokerCatalog).Uuid -RunAsynchronously
```

Create and start a task with a single test suite and a catalog target object, and return immediately not waiting for the tests to complete.

----- EXAMPLE 6 -----

```
$adAccountPool = Get-AcctIdentityPool
$singleTestTaskWithNoObjectDiscovery = Start-EnvTestTask -IgnoreRelatedObjects -TestId ADIdentity_IdentityPool_ValidatePoolIsUnlocked -TargetIdType IdentityPool -TargetId $adAccountPool.IdentityPoolUid
```

Create and start a task with a single test, a target object for that test, and no object discovery based on that target.

----- EXAMPLE 7 -----

```
$singleSimpleTestSuiteTaskTarget = New-EnvTestDiscoveryTargetDefinition -TestSuiteId Infrastructure
$singleTestSuiteTaskTarget = New-EnvTestDiscoveryTargetDefinition -TestSuiteId Catalog -TargetIdType Catalog -TargetIdId $(Get-BrokerCatalog).Uuid
```

Create two different discovery target definitions, put them in an array, then start a task based on both via -InputObject.
Stop-EnvTestTask

Oct 10, 2013
Stops a still running task from completing.

Syntax

Stop-EnvTestTask [-TaskId <Guid>] [-AdminAddress <String>] [CommonParameters]
Stop-EnvTestTask [-Task <EnvTestTask>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Stops a still running task from completing. A task may still be retrieved via Get-EnvTestTask until it Remove-EnvTestTask is called with its task id.

Related topics

Get-EnvTestDefinition
Get-EnvTestSuiteDefinition
Get-EnvTestTask
New-EnvTestTask
Start-EnvTestTask
Switch-EnvTestTask
Remove-EnvTestTask
Add-EnvTestTaskMetadata
Remove-EnvTestTaskMetadata

Parameters

-TaskId <Guid>
The id of the task to stop, retrieveable from the Task.TaskId property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-Task <EnvTestTask>

An EnvTestTask representing the task to stop

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
Default Value | Accept Pipeline Input?
---|---
true (ByValue)

-AdminAddress $String$

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Examples

-------------------------- EXAMPLE 1 --------------------------
Stop-EnvTestTask
Stops the current task from completing if it is still running.

-------------------------- EXAMPLE 2 --------------------------
Stop-EnvTestTask -TestId 50A4139F-2B55-4A97-A1BE-20EE4E124AA3
Stops a task from completing via its id, which is available from an existing task's $Task.TaskId property.

-------------------------- EXAMPLE 3 --------------------------
$secondTask = $(Get-EnvTestTask -List)[1]
Stop-EnvTestTask -Task $secondTask
Stops the second task in the list returned by Get-EnvTestTask -List.
Switch-EnvTestTask

Oct 10, 2013
Sets the current task that will be returned by a call to Get-EnvTestTask with no parameters.

Syntax

Switch-EnvTestTask [-TaskId <Guid>] [-AdminAddress <String>] [CommonParameters]

Switch-EnvTestTask [-Task <EnvTestTask>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Sets the current task that will be returned by a call to Get-EnvTestTask with no parameters.

Related topics

Get-EnvTestDefinition
Get-EnvTestSuiteDefinition
Get-EnvTestTask
New-EnvTestTask
Start-EnvTestTask
Stop-EnvTestTask
Remove-EnvTestTask
Add-EnvTestTaskMetadata
Remove-EnvTestTaskMetadata

Parameters

-TaskId<Guid>

Specifies the identifier of the task to be made current.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-Task<EnvTestTask>

The task object to be made current, retrieveable from the $task.TaskId property.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

System.Management.Automation.PSObject Objects containing the TaskId parameter can be piped to the Remove-EnvTestTask command.

Examples

-------------------------- EXAMPLE 1 --------------------------

Switch-EnvTestTask -TaskId 50A4139F-2B55-4A97-A1BE-20EE4E124AA3
Switches the current task to another via its id, which is available from an existing task's $task.TaskId property.

-------------------------- EXAMPLE 2 --------------------------

$secondTask = $(Get-EnvTestTask -List)[1]
Switch-EnvTestTask -Task $switchTask
Switches the current task to the second in the list returned by Get-EnvTestTask -List.
Test-EnvTestDBConnection

Oct 10, 2013
Tests a database connection for the EnvTest Service.

Syntax
Test-EnvTestDBConnection [-DBConnection] <String> [-LoggingId <Guid>] [ -AdminAddress <String>] [ <CommonParameters>]

Detailed Description
Tests a connection to the database in which the EnvTest Service can store its state. The service will attempt to connect to the database without affecting the current connection to the database.

You do not have to clear the connection to use this command.

Related topics
Get-EnvTestServiceStatus
Get-EnvTestDBConnection
Set-EnvTestDBConnection

Parameters
-DBConnection <String>
Specifies the database connection string to be tested by the EnvTest Service.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
The Test-EnvTestDBConnection command returns an object containing the status of the EnvTest Service if the connection string of the specified data store were to be set to the string being tested, together with extra diagnostics information for the specified connection string.

**DBRejectedConnection**

The database rejected the logon attempt from the EnvTest Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**

The expected stored procedures are missing from the database. This may be because the EnvTest Service schema has not been added to the database.

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**

The version of the EnvTest Service currently in use is incompatible with the version of the EnvTest Service schema on the database. Upgrade the EnvTest Service to a more recent version.

**DBOlderVersionThanService**

The version of the EnvTest Service schema on the database is incompatible with the version of the EnvTest Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**

A database schema upgrade is currently in progress.

**OK**

The Set-EnvTestDBConnection command would succeed if it were executed with the supplied connection string.

**Failed**

The EnvTest Service has failed.

**Unknown**

The status of the EnvTest Service cannot be determined.

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

-----------

**InvalidDBConnectionString**

The database connection string has an invalid format.

**PermissionDenied**

You do not have permission to execute this command.

**AuthorizationError**

There was a problem communicating with the Citrix Delegated Administration Service.

**ConfigurationLoggingError**

The operation could not be performed because of a configuration logging error.
CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS> Test-EnvTestDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Tests a database connection string for the EnvTest Service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS> Test-EnvTestDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Tests an invalid database connection string for the EnvTest Service.
Citrix.Host.Admin.V2

Oct 10, 2013

Overview

### Name | Description
--- | ---
HypHostSnapin | The Host Service PowerShell snap-in provides administrative functions for Hyp Filtering.

### Cmdlets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-HypHostingUnitMetadata</td>
<td>Adds metadata on the given HostingUnit.</td>
</tr>
<tr>
<td>Add-HypHostingUnitNetwork</td>
<td>Makes additional hypervisor networks available for use in a HostingUnit.</td>
</tr>
<tr>
<td>Add-HypHostingUnitStorage</td>
<td>Adds storage locations to a hosting unit.</td>
</tr>
<tr>
<td>Add-HypHypervisorConnectionAddress</td>
<td>Add a connection address to a hypervisor connection.</td>
</tr>
<tr>
<td>Add-HypHypervisorConnectionMetadata</td>
<td>Adds metadata on the given HypervisorConnection.</td>
</tr>
<tr>
<td>Add-HypHypervisorConnectionScope</td>
<td>Add the specified HypervisorConnection(s) to the given scope(s).</td>
</tr>
<tr>
<td>Add-HypMetadata</td>
<td>Adds metadata to a hypervisor connection or a hosting unit.</td>
</tr>
<tr>
<td>Get-HypConfigurationDataForItem</td>
<td>Retrieves the configuration data for an item in the Host Service provider path. Note: For this release, only VM items are supported for this operation.</td>
</tr>
<tr>
<td>Get-HypConfigurationObjectForItem</td>
<td>Retrieves the configuration data for an item in the Host Service provider path. Note: For this release, only VM items are supported for this operation.</td>
</tr>
<tr>
<td>Get-HypDBConnection</td>
<td>Gets the database string for the specified data store used by the Host Service.</td>
</tr>
<tr>
<td>Get-HypDBSchema</td>
<td>Gets a script that creates the Host Service database schema for the specified data store.</td>
</tr>
<tr>
<td>Get-HypDBVersionChangeScript</td>
<td>Gets a script that updates the Host Service database schema.</td>
</tr>
<tr>
<td>Get-HypHypervisorPlugin</td>
<td>Gets the available hypervisor types.</td>
</tr>
<tr>
<td>Get-HypInstalledDBVersion</td>
<td>Gets a list of all available database schema versions for the Host Service.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Get-HypScopedObject</td>
<td>Gets the details of the scoped objects for the Host Service.</td>
</tr>
<tr>
<td>Get-HypService</td>
<td>Gets the service record entries for the Host Service.</td>
</tr>
<tr>
<td>Get-HypServiceAddedCapability</td>
<td>Gets any added capabilities for the Host Service on the controller.</td>
</tr>
<tr>
<td>Get-HypServiceInstance</td>
<td>Gets the service instance entries for the Host Service.</td>
</tr>
<tr>
<td>Get-HypServiceStatus</td>
<td>Gets the current status of the Host Service on the controller.</td>
</tr>
<tr>
<td>Get-HypVMMacAddress</td>
<td>Retrieves a list the MAC addresses for the VMs in the specified connection.</td>
</tr>
<tr>
<td>Get-HypXenServerAddress</td>
<td>Gets all the available addresses for a XenServer hypervisor connection.</td>
</tr>
<tr>
<td>New-HypVMSnapshot</td>
<td>Creates a new snapshot for the specified VM item path.</td>
</tr>
<tr>
<td>Remove-HypHostingUnitMetadata</td>
<td>Removes metadata from the given HostingUnit.</td>
</tr>
<tr>
<td>Remove-HypHostingUnitNetwork</td>
<td>Removes network from a hosting unit.</td>
</tr>
<tr>
<td>Remove-HypHostingUnitStorage</td>
<td>Removes storage from a hosting unit.</td>
</tr>
<tr>
<td>Remove-HypHypervisorConnectionAddress</td>
<td>Removes addresses from the list of available connection addresses.</td>
</tr>
<tr>
<td>Remove-HypHypervisorConnectionMetadata</td>
<td>Removes metadata from the given HypervisorConnection.</td>
</tr>
<tr>
<td>Remove-HypHypervisorConnectionScope</td>
<td>Remove the specified HypervisorConnection(s) from the given scope(s).</td>
</tr>
<tr>
<td>Remove-HypMetadata</td>
<td>Removes metadata from a hypervisor connection or hosting unit.</td>
</tr>
<tr>
<td>Remove-HypServiceMetadata</td>
<td>Removes metadata from the given Service.</td>
</tr>
<tr>
<td>Reset-HypServiceGroupMembership</td>
<td>Reloads the access permissions and configuration service locations for the Host Service.</td>
</tr>
<tr>
<td>Set-HypAdminConnection</td>
<td>Set the controller that will be used by the cmdlets that form the Host Service PowerShell snap-in.</td>
</tr>
<tr>
<td>Set-HypDBConnection</td>
<td>Configures a database connection for the Host Service.</td>
</tr>
<tr>
<td>Set-HypHostingUnitMetadata</td>
<td>Adds or updates metadata on the given HostingUnit.</td>
</tr>
<tr>
<td>Set-HypHostingUnitStorage</td>
<td>Sets options for a storage location on a hosting unit.</td>
</tr>
<tr>
<td>Set-HypHypervisorConnectionMetadata</td>
<td>Adds or updates metadata on the given HypervisorConnection.</td>
</tr>
<tr>
<td>Set-HypServiceMetadata</td>
<td>Adds or updates metadata on the given Service.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>Start-HypVM</td>
<td>Starts a VM.</td>
</tr>
<tr>
<td>Stop-HypVM</td>
<td>Stops a VM by issuing a Shutdown request.</td>
</tr>
<tr>
<td>Test-HypDBConnection</td>
<td>Tests a database connection for the Host Service.</td>
</tr>
<tr>
<td>Test-HypHostingUnitNameAvailable</td>
<td>Checks to see if the proposed name for a hosting unit is unused.</td>
</tr>
<tr>
<td>Test-HypHypervisorConnectionNameAvailable</td>
<td>Checks to see if the proposed name for a hypervisor connection is unused.</td>
</tr>
<tr>
<td>Update-HypHypervisorConnection</td>
<td>Requests the host service to update the connection properties that depend on the version of hypervisor in use.</td>
</tr>
</tbody>
</table>
about_HypHostSnapin

Oct 10, 2013

TOPIC

about_HypHostSnapin

SHORT DESCRIPTION

The Host Service PowerShell snap-in provides administrative functions for the Host Service.

COMMAND PREFIX

All commands in this snap-in have the noun prefixed with 'Hyp'.

LONG DESCRIPTION

The Host Service PowerShell snap-in enables both local and remote administration of the Host Service. It lets you configure XenDesktop deployments to make use of hypervisors, networks, and storage, and enables browsing of their contents using the Host PowerShell provider (Citrix.HypervisorProvider).

The provider creates a default PSDrive with a drive identifier of 'XDHyp'. This drive provides two root folders:

HostingUnits Folder

Contains a list of all the hosting units that have been defined using the new-Item provider command.

Hosting units define not only a hypervisor connection, but also reference networks and storage. Hosting units are used by the Machine Creation Service to provide the information required to create and manage virtual machines that can be used by other services. A hosting unit references a root path. This is a specific point in the provider connection tree. The hosting unit is constrained to provide only items below this point in the tree, which restricts the locations that the Machine Creation Service can use to create virtual machines and the networks and storage that can be used.

Connections Folder

Contains a list of all the hosting unit connections that are defined using the new-Item provider command.

Change directory to a specific connection and use the dir/Get-ChildItem command to list all of the infrastructure (such as folders, hypervisors, networks, storage, and virtual machines) that is available in the hosting unit to which the connection refers. The paths to these items
are used as the input to commands in the Host Service and Machine Creation Service snap-ins.

The contents of the Hosting Unit and Connection folders reflect the content and structure of the hypervisor to which they refer. The item extensions reflect the object type for each item. Not all item types are appropriate for all hypervisor types. Possible item types are:

- Virtual Machine (.vm)
- Snapshot (.snapshot)
- Cluster (.cluster)
- Host (.host)
- HostGroup (.hostgroup)
- DataCenter (.datacenter)
- Folder (.folder)
- ResourcePool (.resourcepool)
- ComputeResource (.computeresource)

When used with a path that refers to the Host provider (with a default drive of 'XDHyp:'), the Host provider extends the standard New-Item, Get-Item, Get-ChildItem, Remove-Item, Rename-Item, and Set-Item commands as described below. For more information about the basic behavior of these commands, see the help for the command. The Move-Item and Copy-Item commands are not supported for the Host provider.

New-Item

The following parameters are available when using New-Item in the Connections directory (or a path that refers to the directory). The Credential parameter is not supported.

- **-Name**
  Specifies the name of the connection, which must not contain any of the following characters: \/;:#.*?=<>|[](){}"'. The Name parameter is optional but, if not included, the connection name must be specified as part of the Path parameter.

- **-HypervisorAddress <String[]>**
  Specifies an array of addresses that can be used to contact the required hypervisor. All the addresses are considered equivalent, that is, all of the addresses provide access to the same virtual machines, snapshots, network, and storage.

- **-ConnectionType <ConnectionType>**
  Specifies the type of hypervisor that the connection is for.
  Supported hypervisor types are:

  XenServer
SCVMM (Microsoft Hyper-V)
vCenter (VMware vSphere/ESX)
Custom

-**PluginId <String>**
  Specifies the class name for the Citrix Managed Machine library that is used to access the hypervisor. This list can be obtained using the Get-HypHypervisorPlugin command. The PluginId parameter must be specified if the ConnectionType is set to 'Custom'.

-**UserName <String>, -Password <String>, -SecurePassword <SecureString>**
  Specifies the credentials for the connection to the hypervisor. The password can be given as either Password or SecurePassword, but not both. The UserName parameter and either Password or SecurePassword specify the same information as the HypervisorCredential parameter, so use of one precludes use of the other.

-**HypervisorCredential <PSCredential>**
  Specifies credentials for the connection to the hypervisor. The HypervisorCredential parameter specifies the same information as UserName and either Password or SecurePassword, so use of one precludes use of the other.

-**Persist**
  Specifies that the connection details are persistent. If this parameter is not included, the connection is only held for the duration of the current runspace and PSDrive combination. Only persistent connection items are visible to administrators in other runspaces and PSDrives. Hosting units cannot be created from connections that are not persistent.

-**AdminAddress <String>**
  Specifies the address of the Host Service that the command communicates with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command attempts to use a local Host Service.

-**LoggingId <String>**
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-**Scopes <String[]>**
Specifies the list of administrative scopes that the new connection will be a part of. The scopes will control which administrators are able to work with the connection.

The following parameters are available when using New-Item in the HostingUnits directory (or a path that refers to the directory).

- **Name**
  Specifies the name of the hosting unit, which must not contain any of the following characters: \/:.*?<>|(). The Name parameter is optional but, if not included, the hosting unit name must be provided as part of the Path parameter.

- **HypervisorConnectionName <String>**
  Specifies the name of the connection that the hosting unit uses. To create a hosting unit, the referenced connection must be persistent and the ConnectionType must not be set to 'Custom'.

- **HypervisorConnectionUid <Guid>**
  Specifies the unique identifier of the connection that the hosting unit uses. To create a hosting unit, the referenced connection must be persistent and the ConnectionType must not be set to 'Custom'.

- **RootPath <String>**
  Specifies the location in a connection that is used as the starting reference for the hosting unit. The path must point to an item in the connection that is marked as a SymLink. The root of a XenServer connection is a special case that is also considered a SymLink. If this parameter is not included, the current location in the provider is used.

- **NetworkPath <String>**
  Specifies the path in a connection to the network item that is used when the Machine Creation Service creates new virtual machines.

- **StoragePath <String>**
  Specifies one or more paths in a connection to storage items that are used when the Machine Creation Service creates new virtual machines. After they are set, storage paths can be modified using the Add-HypHostingUnitStorage and Remove-HypHostingUnitStorage commands.

- **PersonalvDiskStoragePath <String>**
  Specifies one or more paths in a connection to storage items that are used when the Machine Creation Service creates disks for the virtual machines. After they are set, storage paths can be modified.
using the Add-HypHostingUnitStorage and Remove-HypHostingUnitStorage commands.

-NoVmTagging
  Specifies that new virtual machines are not tagged with metadata from the hypervisor. By default, all virtual machines created by the Machine Creation Service are tagged to show they are created by XenDesktop. These tags are used by the provider to restrict the list of virtual machines displayed when viewing the content of the Connections or HostingUnit paths. If this parameter is not included, all virtual machines are displayed at all times.

-AdminAddress <String>
  Specifies the address of the Host Service that the command will communicate with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command attempts to use a local Host Service.

-UseLocalStorageCaching
  When Get-HypServiceAddedCapability indicates that the LocalStorageCaching feature is available this parameter can be set to specify that the virtual machines created for this hosting unit will use local storage caching for their disk images.

-GpuGroup

  Specifies a path to a GPU Group in a connection that will be used when Machine Creation Services creates VMs. Only a single GPU Group is supported and the value is immutable.

-LoggingId <String>
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Get-Item
Get-ChildItem (alias dir)
Rename-Item
Remove-Item
The following additional parameters are available.

**AdminAddress**
Specifies the address of the Host Service that the command communicates with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command attempts to use a local Host Service.

- **-LoggingId <String>**
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

The Path parameter can be specified by name or by using the unique identifier for the connection or hosting unit enclosed in braces. For example:

- **-Path XDHyp:/{1233-3213ACDF-12323}**

The Filter parameter is not supported. Virtual machines created by the Machine Creation Service are only returned if the -Force parameter is used or if virtual machine tagging was disabled on the hosting unit with the NoVmTagging parameter when the VMs were created.

---

**Set-Item**

The following parameters are available when using Set-Item in the Connections directory.

- **-UserName <String>, -Password <String>, -SecurePassword <SecureString>**
  Specifies the credentials for the connection to the hypervisor. The password can be given as either Password or SecurePassword, but not both. The UserName parameter and either Password or SecurePassword specify the same information as the HypervisorCredential parameter, so use of one precludes use of the other.

- **-HypervisorCredential <PSCredential>**
  Specifies credentials for the connection to the hypervisor. The HypervisorCredential parameter specifies the same information as the UserName parameter and either Password or SecurePassword, so use of one precludes use of the other.
-HypervisorAddress <string[]>
   Specifies the addresses of the hypervisor that this connection represents.
   This replaces all existing addresses.

-LoggingId <String>
   Specifies the identifier of the high level operation that this cmdlet call
   forms a part of. Desktop Studio and Desktop Director typically create
   High Level Operations. PowerShell scripts can also wrap a series of cmdlet
   calls in a High Level Operation by way of the Start-LogHighLevelOperation
   and Stop-LogHighLevelOperation cmdlets.

-MaintenanceMode <Boolean>
   Places the connection into maintenance mode which will disable all communication
   between XenDesktop and the Hypervisor. Use this mode when making changes to the
   hypervisor, for instance if the password for the access account needs to be changed.

-AdminAddress <String>
   Specifies the address of the Host Service that the command
   communicates with. After it is set, this address is used for all commands
   in the Host Service PowerShell snap-in. If this parameter is not
   included or set by another command, or if Set-HypAdminConnection
   has not been used, the command tries to use a local Host
   Service.

   The Credential parameter is not supported. Addresses can be modified
   with the Add-HypHypervisorConnectionAddress and
   Remove-HypHypervisorConnectionAddress commands. Metadata can be
   modified with the Add-HypMetadata and Remove-HypMetadata commands.

The following parameters are available when using New-Item in the
HostingUnits directory.

-Name
   Specifies the name of the hosting unit, which must not contain any
   of the following characters: V;#.*?<>\[\]()"'. The Name
   parameter is optional but, if not included, the hosting unit name
   must be provided as part of the Path parameter.

-NetworkPath <String>
   Specifies the path in a connection to the network item that
   is used when the Machine Creation Service creates new virtual
   machines.

-NoVmTagging
   Specifies that new virtual machines are not tagged with
   metadata from the hypervisor. By default, all virtual machines
   created by the Machine Creation Service are tagged to show they are created
by XenDesktop. These tags are used by the provider to restrict the list of virtual machines displayed when viewing the content of the Connections or HostingUnit paths. If this parameter is not included, all virtual machines are displayed at all times.

-AdminAddress <String>
  Specifies the address of the Host Service that the command communicates with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command attempts to use a local Host Service.

-UseLocalStorageCaching <bool>
  When Get-HypServiceAddedCapability indicates that the LocalStorageCaching feature is available this parameter can be set to specify that the virtual machines created for this hosting unit will use local storage caching for their disk images.

-LoggingId <String>
  Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Storage can be modified with the Add-HypHostingUnitStorage and Remove-HypHostingUnitStorage commands. Metadata can be modified with the Add-HypMetadata and Remove-HypMetadata commands.

The item types returned when these commands are used in the Host Service provider are defined in the Object Definitions section below.

EXAMPLES

To Create a Persistent Hypervisor Connection

```powershell
new-item -path "xdhyp:\Connections" -Name MyConn -ConnectionType XenServer -HypervisorAddress "http:\address" -UserName user -Password password -Persist
```

```powershell
PSPath: Citrix.HostingUnitService.Admin.V1.0\Citrix.Hypervisor::XDHyp:\Connections\MyConn
PSParentPath: Citrix.HostingUnitService.Admin.V1.0\Citrix.Hypervisor::XDHyp:\Connections
PSChildName: MyConn
PSDrive: XDHyp
```
To Create a Hosting Unit

```
new-item -Path "xdhyp:\HostingUnits"
    -Name MyHU
    -HypervisorConnectionName MyConn
    -RootPath XDHYP:\Connections\MyConn
    -NetworkPath XDHYP:\Connections\MyConn\Network 0.network
    -StoragePath XDHYP:\Connections\MyConn\Local storage on myXenServer.storage
```

Relative paths can be used for all parameters.
OBJECT DEFINITIONS

Citrix.XDInterServiceTypes.HypervisorConnection

The hypervisor connection object is returned when a connection item is located. This item has the following parameters.

HypervisorConnectionUid <Guid>
   Specifies the unique identifier for the connection item.

HypervisorConnectionName <String>
   Specifies the name of the connection item.

ConnectionType <Citrix.XDInterServiceTypes.ConnectionType>
   Specifies the type of hypervisor that the connection is for.
   Supported hypervisor types are:
   - XenServer
   - SCVMM (Microsoft Hyper-V)
   - vCenter (VMware vSphere/ESX)
   - Custom

HypervisorAddress <String[]>
   Specifies the addresses that can be used to contact the required hypervisor.

Username <String>
   Specifies the administrator user name for the connection to the hypervisor (from the user name and password given as administrator credentials when setting up this connection)

Persistent <Boolean>
   Specifies whether or not the connection is persistent.

PluginId <String>
   Specifies the Citrix Managed Machine class identifier for the hypervisor.

SupportsPvsVM <Boolean>
   Specifies whether or not the connection can be used as part of a hosting unit. If this parameter is set to 'True', a hosting unit referencing this connection can be created.

Revision <Guid>
   A unique identifier that is changed every time any properties of the connection are changed.
MaintenanceMode <Boolean>
   Specifies whether or not the connection is currently in maintenance
   mode.

Metadata <Citrix.XDInterServiceTypes.Metadata[]>
   The collection of metadata associated with the connection.

Citrix.XDInterServiceTypes.HostingUnit

   The hosting unit object is returned when a hosting unit item is
   located. This item has the following parameters.

HostingUnitName <String>
   Specifies the name of the hosting unit.

HostingUnitUid <Guid>
   Specifies the unique identifier for the hosting unit.

HypervisorConnection <Citrix.XDInterServiceTypes.HypervisorConnection>
   Specifies the hypervisor connection item that the hosting unit
   references.

NetworkId <String>
   Specifies the unique identifier for the network in the
   hypervisor context that the hosting unit references.

NetworkPath <String>
   Specifies the path to the network in the Host Service provider.

RootId <String>
   Specifies the unique identifier for the root connection item in
   the hypervisor context that the hosting unit references.

RootPath <String>
   Specifies the path to the root of the hosting unit from within the
   Host Service provider.

Storage <Citrix.XDInterServiceTypes.Storage[]>
   Specifies the collection of storage objects that are defined for
   use as part of the hosting unit. This object contains the
   following parameters.

StorageID <String>
   Specifies the identifier for the storage in the hypervisor
   context.
StoragePath <String>
   Specifies the path to the storage in the Host Service provider.

VMTaggingEnabled <Boolean>
   Specifies whether or not virtual machine metadata is used to tag virtual machines created by XenDesktop.

Metadata <Citrix.XDInterServiceTypes.Metadata[]>
   Specifies the collection of metadata associated with the hosting unit.

Citrix.HostingUnitService.Sdk.HypervisorObject

The hypervisor object is returned when any item is located from within a Connection or HostingUnit folder. This item has the following parameters.

AdditionalData <Dictionary<String,String>
   Stores extra untyped data for the item. This dictionary contains different information for each item type.

   For Storage Items
   Key = StorageType
   Values = Shared or Local

   For VM Items
   Key = PowerState
   Values = PoweredOn,
   PoweredOff,
   Suspended,
   TransitioningToOn,
   TransitioningToOff,
   Suspending,
   Resuming,
   Unknown,
   Error

   Name <String>
   Specifies the name of the item.

   FullName <String>
Specifies the name with the appropriate file extension.

ObjectPath <String>
   Specifies the relative path to the item from the root of the
   connection of which the item is part.

FullPath <String>
   Specifies the absolute path to the item in the
   Citrix.Hypervisor provider.

Id <String>
   Specifies the identity of the item in the hypervisor context.

IsContainer <Boolean>
   Specifies whether or not the item can contain other items.

IsSymLink <Boolean>
   Specifies whether or not the item can be used as a RootPath of a
   hosting unit.

ObjectType <Citrix.HostingUnitService.Sdk.NodeType>
   Specifies the item type.

ERROR CODES

The provider commands can return the following error codes.

New-Item

ConnectionNameOrUidInvalid
   The name or unique identifier of the hypervisor connection
   specified for the hosting unit is invalid.

HostingUnitRootPathInvalid
   The root path specified for the hosting unit is invalid. The root
   path must be a valid item in the hypervisor tree and a SymLink.

HostingUnitNetworkPathInvalid
   The network path specified for the hosting unit is invalid. The
   network path must be a valid item in the hypervisor tree and
   relative to the root path.

HostingUnitStoragePathInvalid
   The storage path specified for the hosting unit is invalid. The
   storage path must be a valid item in the hypervisor tree and
   relative to the root path.
HostingUnitDuplicateObjectExists
A hosting unit object with the same name already exists. Hosting unit names must be unique.

HostingUnitStorageDuplicateObjectExists
A hosting unit storage object with the same storage path already exists for this hosting unit. There can only be one object for each combination of hosting unit and storage path.

HypervisorNotContactable
The hypervisor could not be contacted at the supplied address, which is either invalid or unreachable.

HypervisorAddressInvalidFormat
The address is not in a valid form for the specified hypervisor type.

ConnectionAddressInvalid
The specified address is invalid and does not belong to the same pool.

HypervisorConnectionDuplicateObjectExists
A hypervisor connection object with the same name already exists. Hypervisor connection names must be unique.

HypervisorConnectionAddressDuplicateObjectExists
A hypervisor connection address object with the same address already exists for this hypervisor connection. There can only be one object for each combination of hypervisor connection and address.

HypervisorConnectionForHostingUnitsIsVirtual
The specified hypervisor connection for the hosting unit is a virtual non-persistent connection. A persistent connection is required when creating a hosting unit.

InputNameInvalid
The name specified for the hosting unit or hypervisor connection is invalid because it contains one or more of the following characters: \/.:*?<>|()"'.

InputPathInvalid
The specified path is invalid because it is not in one of the following formats:

XDHyp:\Connections\<Name>
ExceptionThrown
An unexpected error occurred.

DatabaseError
There was a problem communicating with the database.

CommunicationError
There was a problem communicating with the remote service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ScopeNotFound
One or more of the scopes nominated for the new connection do not exist.

CannotCreateSecurityGroupHere
Security groups can only be created in a virtual private cloud (VPC).

Get-Item and Get-ChildItem

HypervisorConnectionObjectNotFound
The hypervisor connection object specified in the path could not be found.

HostingUnitObjectNotFound
The specified hosting unit object could not be found.

HypervisorInMaintenanceMode
The hypervisor for the specified connection is currently in maintenance mode and cannot be accessed.

FailureToRetrieveConnectionPassword
The password for the HypervisorConnection cannot be retrieved or decrypted.

ExceptionThrown
An unexpected error occurred.

DatabaseError
There was a problem communicating with the database.

CommunicationError
There was a problem communicating with the remote service.

Remove-Item
HostingUnitObjectToDeleteDoesNotExist
The specified hosting unit object does not exist.

HypervisorConnectionObjectToDeleteDoesNotExist
The specified hypervisor connection object does not exist.

InputPathInvalid
The specified path is invalid because it is not in one of the following formats:

- XDHyp:\Connections\<Name>
- XDHyp:\Connections\{Guid}
- XDHyp:\HostingUnits\<Name>
- XDHyp:\HostingUnits\{Guid}

HypervisorConnectionObjectToDeleteIsInUse
The specified hypervisor connection object cannot be deleted because it is being used by one or more hosting units.

ExceptionThrown
An unexpected error occurred.

DatabaseError
There was a problem communicating with the database.

CommunicationError
There was a problem communicating with the remote service.

Rename-Item

InputPathInvalid
The specified path is invalid because it is not in one of the following formats:

- XDHyp:\Connections\<Name>
- XDHyp:\Connections\{Guid}
- XDHyp:\HostingUnits\<Name>
- XDHyp:\HostingUnits\{Guid}

HostingUnitDuplicateNameExists
A hosting unit object with the same name already exists. Hosting unit names must be unique.

HypervisorConnectionDuplicateNameExists
A hypervisor connection object with the same name already exists. Hypervisor connection names must be unique.
InputNameInvalid
The new name specified for the hosting unit or hypervisor connection is invalid because it contains one or more of the following characters: \/,:.*?<>\[\]\(){}\'.

ExceptionThrown
An unexpected error occurred.

DatabaseError
There was a problem communicating with the database.

CommunicationError
There was a problem communicating with the remote service.

Set-Item

InputPathInvalid
The specified path is invalid because it is not in one of the following formats:

XDHyp:\Connections\<Name>
XDHyp:\Connections\{Guid}
XDHyp:\HostingUnits\<Name>
XDHyp:\HostingUnits\{Guid}

HostingUnitObjectToUpdateDoesNotExist
The specified hosting unit object does not exist.

HostingUnitNetworkPathInvalid
The new network path specified for the hosting unit is invalid.
Either the network path does not exist or it is not relative to the root path of the hosting unit.

HypervisorConnectionObjectToUpdateDoesNotExist
The specified hypervisor connection object does not exist.

ExceptionThrown
An unexpected error occurred.

DatabaseError
There was a problem communicating with the database.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
DataStoreException
An error occurred in the service while attempting a database operation.
Communication to the database failed for various reasons.

CommunicationError
There was a problem communicating with the remote service.
about_Hyp_Filtering

Oct 10, 2013

TOPIC
XenDesktop - Advanced Dataset Filtering

SHORT DESCRIPTION
Describes the common filtering options for XenDesktop cmdlets.

LONG DESCRIPTION

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get-cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

```
Get-Noun | Where { $_.Size = 'Small' } | Sort 'Date' | Select -First 10
```

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get-cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

```
Get-Noun -Size 'Small' -SortBy 'Date' -MaxRecordCount 10
```

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

- `-MaxRecordCount <int>`

  Specifies the maximum number of results to return.
  For example, to return only the first nine results use:

  ```
  Get-Noun -MaxRecordCount 9
  ```

  If not specified, only the first 250 records are returned, and if more are available, a warning is produced:
WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

To retrieve all records, specify a large number for -MaxRecordCount. As the value is an integer, you can use the following:

Get-<Noun> -MaxRecordCount [int]:MaxValue

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount

....

Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
  + Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
  + CategoryInfo : OperationStopped: ( ) [Get-<Noun>], PartialDataException
  + FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.
Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-sortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values. Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

-sortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'

-filter <string>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter 'Name -like "High*" -or (Priority -eq 1 -and Severity -ge 2)'

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

EXAMPLES

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company "citrix" -Product '[X]EN''
Get-<Noun> -Product "Xen"* -Company "CITRIX"
Get-<Noun> -Filter { Company -eq 'Citrix' -and Product -like 'Xen*' }

https://docs.citrix.com
See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```powershell
Get-<Noun> -Company "Abc[*]"                 # Matches Abc*
Get-<Noun> -Company "Abc**"                   # Matches Abc*
Get-<Noun> -Filter { Company -eq "Abc**" }   # Matches Abc*
Get-<Noun> -Filter { Company -eq "A`"B`'C" } # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```powershell
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

```powershell
Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```powershell
Get-<Noun> -Filter 'Enabled'      # Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled' # Equivalent to 'Enabled -eq $false'
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```powershell
Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:
By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

```
Get-<Noun> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z" }
$d = [DateTime]"2010-08-23T12:30:00.0Z"
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }
```

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

```
Get-<Noun> -Filter { StartTime -ge '-2' }         # Two days ago
Get-<Noun> -Filter { StartTime -ge '-1:30' }      # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge '-0:0:30' }    # 30 seconds ago
```

ARRAY PROPERTIES

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings. Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

```
Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*" }
Get-<Noun> -Filter { Users -contains "Fred*" }
```

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter { User -eq "Frederick" }
# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter { User -lt 'F' }
```

COMPLEX EXPRESSIONS
When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

\[
\text{Get-<Noun> -Filter \{ Shape -eq 'Circle' -or Shape -eq 'Square' \}}
\]

\[
\text{$\text{shapes} = \{'Circle', 'Square'$}}
\]

\[
\text{Get-<Noun> -Filter \{ Shape -in $\text{shapes}$ \}}
\]

\[
\text{$\text{sides} = 1..4$}
\]

\[
\text{Get-<Noun> -Filter \{ Sides -notin $\text{sides}$ \}}
\]

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

\[
\text{Get-<Noun> -Filter \{ Size -gt 4 -or (Color -eq 'Blue' -and Shape -eq 'Circle') \}}
\]

\[
\text{Get-<Noun> -Filter \{ Sides -lt 5 -and -not (Color -eq 'Blue' -and Shape -eq 'Circle') \}}
\]

PAGING

The simplest way to page through data is to use the -Skip and -MaxRecordCount parameters. So, to read the first three pages of data with 10 records per page, use:

\[
\text{Get-<Noun> -Skip 0 -MaxRecordCount 10 <other filtering criteria>}
\]

\[
\text{Get-<Noun> -Skip 10 -MaxRecordCount 10 <other filtering criteria>}
\]

\[
\text{Get-<Noun> -Skip 20 -MaxRecordCount 10 <other filtering criteria>}
\]

You must include the same filtering criteria on each call, and ensure that the data is sorted consistently.

The above approach is often acceptable, but as each call performs an independent query, data changes can result in records being skipped or appearing twice. One approach to improve this is to sort by a unique id field and then start the search for the next page at the unique id after the last unique id of the previous page. For example:

\[
\text{# Get the first page}
\]

\[
\text{Get-<Noun> -MaxRecordCount 10 -SortBy SerialNumber}
\]

\[
\begin{array}{l}
\text{SerialNumber ...} \\
\text{---------- ...} \\
\text{A120004} \\
\text{A120007} \\
\text{... 7 other records ...} \\
\text{A120900}
\end{array}
\]

\[
\text{# Get the next page}
\]

\[
\text{Get-<Noun> -MaxRecordCount 10 -Filter \{ FirstName -gt 'A120900' \}}
\]

\[
\text{SerialNumber ...}
\]
FILTER SYNTAX DEFINITION

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= "{" <ComponentList> "}"

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

    <Component>

<Component> ::= <NotOperator> <Factor> |

    <Factor>

<Factor> ::= "{" <ComponentList> "}" |

    <PropertyName> <ComparisonOperator> <Value> |<PropertyName>

<AndOrOperator> ::= "-and" | "-or"

<NotOperator> ::= "-not" | "!"

<ComparisonOperator>

    ::= "-eq" | "-ne" | "-le" | "-ge" | "-lt" | "-gt" |

    "-like" | "-notlike" | "-contains" | "-notcontains" |

    "-in" | "-notin"

(PropertyName) ::= <simple name of property>

(Value) ::= <string literal> | <numeric literal> |

    <scalar variable> | <array variable> |

    "$null" | "$true" | "$false"
Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30" means 1 hour and 30 minutes ago.
Add-HypHostingUnitMetadata

Oct 10, 2013
Adds metadata on the given HostingUnit.

Syntax
Add-HypHostingUnitMetadata [-HostingUnitUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Add-HypHostingUnitMetadata [-HostingUnitUid] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Add-HypHostingUnitMetadata [-HostingUnitName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Add-HypHostingUnitMetadata [-HostingUnitName] <String> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Add-HypHostingUnitMetadata [-InputObject] <HostingUnit[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Add-HypHostingUnitMetadata [-InputObject] <HostingUnit[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Provides the ability for additional custom data to be stored against given HostingUnit objects. This cmdlet will not overwrite existing metadata on an object - use the Set-HypHostingUnitMetadata cmdlet instead.

Related topics
Set-HypHostingUnitMetadata
Remove-HypHostingUnitMetadata

Parameters
-HostingUnitUid <Guid>
Id of the HostingUnit

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-HostingUnitName <String>
Name of the HostingUnit

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
</table>
### Default Value

| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

### InputObject <HostingUnit[]>

Objects to which the metadata is to be added.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

### Map <PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with `new-object "System.Collections.Generic.Dictionary[String,String]"`).

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

### Name <String>

Specifies the property name of the metadata to be added. The property must be unique for the HostingUnit specified. The property cannot contain any of the following characters `/;#?*<>|[]"'`.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

### Value <String>

Specifies the value for the property.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

### LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

---

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.2044
**-AdminAddress <String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**

Citrix.Host.Sdk.Metadata

Add-HypHostingUnitMetadata returns an array of objects containing the new definition of the metadata.

```
\nProperty <string>
\n\n\nProperty <string>
\n\n\nSpecifies the name of the property.
\nValue <string>
\n\n\nSpecifies the value for the property.
```

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

--------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DuplicateObject

One of the specified metadata already exists.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Add-HypHostingUnitMetadata -HostingUnitUid 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the HostingUnit with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Add-HypHostingUnitNetwork

Oct 10, 2013
Makes additional hypervisor networks available for use in a HostingUnit.

Syntax
Add-HypHostingUnitNetwork [-LiteralPath] <String> [-NetworkPath] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
This command lets you extend the set of hypervisor networks that are made available through the HostingUnit to the Citrix Machine Creation Services. When new machines are created, their virtual NICs can be associated only with networks that are in this set. This command cannot be used if the connection for the hosting unit is in maintenance mode.

Related topics
New-Item
Add-HypMetadata
remove-HypHostingUnitNetwork

Parameters
-LiteralPath <String>
Specifies the path to the hosting unit to which network will be added. The path must be in one of the following formats: <drive>:\HostingUnits\<HostingUnitName> or <drive>:\HostingUnits\{<HostingUnit Uid>}

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-NetworkPath <String>
Specifies the path to the network that will be added. The path must be in one of the following formats: <drive>:\Connections\<ConnectionName>\MyNetwork.network or <drive>:\Connections\<Connection Uid>\MyNetwork.network or <drive>:\HostingUnits\<HostingUnitName>\MyNetwork.network or <drive>:\HostingUnits\{<hostingUnit Uid>}\MyNetwork.network

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snap-in connects. This can be a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
Default Value | LocalHost. When a value is provided by any cmdlet, this value becomes the default.
--- | ---
Accept Pipeline Input? | false

### Input Type

- **System.String** You can pipe a string that contains a path to Add-HypHostingUnitNetwork (NetworkPath parameter).

### Return Values

- **Citrix.Host.Sdk.HostingUnit**

  Add-HypHostingUnitNetwork returns an object containing the new definition of the hosting unit.

- **HostingUnitUid <Guid>**
  Specifies the unique identifier for the hosting unit.

- **HostingUnitName <string>**
  Specifies the name of the hosting unit.

- **HypervisorConnection <Citrix.Host.Sdk.HypervisorConnection>**
  Specifies the connection that the hosting unit uses to access a hypervisor.

- **RootId <string>**
  Identifies, to the hypervisor, the root of the hosting unit.

- **RootPath <string>**
  The hosting unit provider path that represents the root of the hosting unit.

- **Storage <Citrix.Host.Sdk.Storage[]>**
  The list of storage items that the hosting unit can use.

- **PersonalvDiskStorage <Citrix.XDPowerShell.Storage[]>**
  The list of storage items that the hosting unit can use for storing personal data.

- **VMTaggingEnabled <Boolean>**
  Specifies whether the metadata in the hypervisor can be used to store information about the XenDesktop Machine Creation Services.

- **NetworkId <string>**
  The hypervisor's internal identifier that represents the default network specified for the hosting unit.

- **NetworkPath <string>**
  The hosting unit provider path to the default network specified for the hosting unit.

- **Metadata <Citrix.Host.Sdk.Metadata[]>**
  A list of key value pairs that can store additional information about the hosting unit.

- **PermittedNetworks <Citrix.Host.Sdk.Network[]>**
  A full list of the hypervisor networks that are exposed for use in the hosting unit.

### Notes

- The network path must be valid for the hosting unit. The rules that are applied are as follows:
  - XenServer (HypervisorConnection Type = XenServer)
  - VMware vSphere/ESX (HypervisorConnection Type = vCenter)

  The network path must be directly contained in the root path item of the hosting unit.
Microsoft SCVMM/Hyper-v (HypervisorConnection Type = SCVMM)

NA

In the case of failure, the following errors can be produced.

Error Codes
-------------

HostingUnitsPathInvalid
The path provided is not to an item in a subdirectory of a hosting unit item.

HostingUnitNetworkPathInvalid
The specified path is invalid.

HostingUnitNetworkPathInvalid
The network path cannot be found or is invalid. See notes above about validity.

HostingUnitNetworkDuplicateObjectExists
The specified network path is already part of the hosting unit.

HypervisorInMaintenanceMode
The hypervisor for the connection is in maintenance mode.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation. Communication to the database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples
----------------------- EXAMPLE 1 -----------------------

c:\PS> Add-HypHostingUnitNetwork -LiteralPath 'XDHyp:\HostingUnits\MyHostingUnit' -NetworkPath 'XDHyp:\HostingUnits\MyHostingUnits\newNetwork.network'

HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
HostingUnitName      : MyHostingUnit
HypervisorConnection : MyConnection
RootPath             : /
RootId               :
NetworkPath          : /Network 0.network
NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
Storage              : {/Local storage.storage}
PersonalvDiskStorage : {}
VMTaggingEnabled : True
Metadata : {}
PermittedNetworks : {/Network 0.network, /newNetwork.network}

The command adds a new network called "newNetwork.network" to the hosting unit called "MyHostingUnit".

-------------------------- EXAMPLE 2 --------------------------

```bash
XDHyp:\HostingUnits\MyHostingUnit>Add-HypHostingUnitNetwork -LiteralPath . -NetworkPath 'XDHyp:\HostingUnits\MyHostingUnits\newNetwork.network'
```

HostingUnitUid : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
HostingUnitName : MyHostingUnit
HypervisorConnection : MyConnection
RootPath : /
RootId :
NetworkPath : /Network 0.network
NetworkId : ab47080b-ca15-771a-c8dc-6ad9650158f1
Storage : {/Local storage.storage}
PersonalvDiskStorage : {}
VMTaggingEnabled : True
Metadata : {}
PermittedNetworks : {/Network 0.network, /newNetwork.network}

The command adds a new network called "newNetwork.network" to the current directory. The dot (.) represents the current location (not its contents).

-------------------------- EXAMPLE 3 --------------------------

```bash
XDHyp:\HostingUnits\MyHostingUnit>dir *.network | Add-HypHostingUnitNetwork -LiteralPath XDHyp:\HostingUnits\MyHostingUnits
```

The command adds all of the networks that are available in the hosting unit to the specified hosting unit.

-------------------------- EXAMPLE 4 --------------------------

```bash
c:\PS>Add-HypHostingUnitNetwork -LiteralPath XDHyp:\HostingUnits\MyHostingUnit -NetworkPath 'XDHyp:\HostingUnits\MyHostingUnits\newNetwork.network'
```

HostingUnitUid : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
HostingUnitName : MyHostingUnit
HypervisorConnection : MyConnection
RootPath : /
RootId :
NetworkPath : /Network 0.network
NetworkId : ab47080b-ca15-771a-c8dc-6ad9650158f1
Storage : {/Local storage.storage}
PersonalvDiskStorage : {}
VMTaggingEnabled : True
Metadata : {}
PermittedNetworks : {/Network 0.network, /newNetwork.network}

The command adds a new network location called "newNetwork.network" to the hosting unit called "MyHostingUnit".
Add-HypHostingUnitStorage

Oct 10, 2013

Adds storage locations to a hosting unit.

Syntax

Add-HypHostingUnitStorage [-LiteralPath] <String> [-StoragePath] <String> [-StorageType <StorageType>] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

This command lets you add storage locations that can store the hard disks required by the virtual machines created by the Citrix Machine Creation Services. This command cannot be used if the connection for the hosting unit is in maintenance mode.

Related topics

New-item

Add-HypMetadata

remove-HypHostingUnitStorage

Parameters

-LiteralPath <String>

Specifies the path to the hosting unit to which storage will be added. The path must be in one of the following formats: \HostingUnits\HostingUnitName or \HostingUnits\{HostingUnit Uid\}

Required? true

Default Value

Accept Pipeline Input? false

-StoragePath <String>

Specifies the path to the storage that will be added. The path must be in one of the following formats: \Connections\ConnectionName\MyStorage.storage or \Connections\{Connection Uid\}\MyStorage.storage or \HostingUnits\HostingUnitName\MyStorage.storage or \HostingUnits\{hostingUnit Uid\}\MyStorage.storage

Required? true

Default Value

Accept Pipeline Input? true (ByValue)

-StorageType <StorageType>

Specifies the type of storage in StoragePath. Supported storage types are: OSStorage PersonalDiskStorage

Required? false

Default Value OSStorage

Accept Pipeline Input? false

-LoggingId <Guid>

 Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Required? false

Default Value

Accept Pipeline Input? false

-AdminAddress <String>

 Specifies the address of a XenDesktop controller that the PowerShell snap-in connects. This can be a host name or an IP address.

Required? false

Default Value localhost. When a value is provided by any cmdlet, this value becomes the default.

Accept Pipeline Input? false

Input Type

System.String You can pipe a string that contains a path to Add-HypHostingUnitStorage (StoragePath parameter).
Return Values

Citrix.Host.Sdk.HostingUnit

Add-HypHostingUnitStorage returns an object containing the new definition of the hosting unit.

HostingUnitUId <Guid>

Specifies the unique identifier for the hosting unit.

HostingUnitName <string>

Specifies the name of the hosting unit.

HypervisorConnection <Citrix.Host.Sdk.HypervisorConnection>

Specifies the connection that the hosting unit uses to access a hypervisor.

RootId <string>

Identifies, to the hypervisor, the root of the hosting unit.

RootPath <string>

The hosting unit provider path that represents the root of the hosting unit.

Storage <Citrix.Host.Sdk.Storage[]>

The list of storage items that the hosting unit can use.

PersonalvDiskStorage <Citrix.XDPowerShell.Storage[]>

The list of storage items that the hosting unit can use for storing personal data.

VMTaggingEnabled <Boolean>

Specifies whether the metadata in the hypervisor can be used to store information about the XenDesktop Machine Creation Services.

NetworkId <string>

The hypervisor's internal identifier that represents the network specified for the hosting unit.

NetworkPath <string>

The hosting unit provider path to the network specified for the hosting unit.

Metadata <Citrix.Host.Sdk.Metadata[]>

A list of key value pairs that can store additional information about the hosting unit.

Notes

The storage path must be valid for the hosting unit. The rules that are applied are as follows: XenServer (HypervisorConnection Type = XenServer)

NA

VMWare vSphere/ESX (HypervisorConnection Type = vCenter)

The storage path must be directly contained in the root path item of the hosting unit.

Microsoft SCVMM/Hyper-v (HypervisorConnection Type = SCVMM)

Only one storage entry for these connection types is valid, and it must reference an SMB share. Additionally, if a Hyper-V failover cluster is used the SMB share must be the top-level mount point of the cluster shared volume on one of the servers in the cluster (i.e. C:\ClusterStorage).

In the case of failure, the following errors can be produced.

Error Codes

----------

HostingUnitsPathInvalid

The path provided is not to an item in a subdirectory of a hosting unit item.

HostingUnitStoragePathInvalid

The specified path is invalid.

HostingUnitStoragePathInvalid

The storage path cannot be found or is invalid. See notes above about validity.

HostingUnitStorageDuplicateObjectExists

The specified storage path is already part of the hosting unit.

HypervisorInMaintenanceMode
The hypervisor for the connection is in maintenance mode.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation. Communication to the database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS> Add-HypHostingUnitStorage -LiteralPath XDHyp:HostingUnits\MyHostingUnit -StoragePath 'XDHyp:HostingUnits\MyHostingUnits\newStorage.storage'
HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
HostingUnitName      : MyHostingUnit
HypervisorConnection : MyConnection
RootPath             : /
RootId               :
NetworkPath          : /Network 0.network
NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
Storage              : {/Local storage.storage, /newStorage.storage}
PersonalDiskStorage : {/newStorage.storage}
VMTaggingEnabled     : True
Metadata             : {}
The command adds a new storage location called "newStorage.storage" to the hosting unit called "MyHostingUnit".

-------------------------- EXAMPLE 2 --------------------------
XDHyp:HostingUnits\MyHostingUnit>Add-HypHostingUnitStorage -LiteralPath . -StoragePath 'XDHyp:HostingUnits\MyHostingUnits\newStorage.storage' -StorageType OSStorage
HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
HostingUnitName      : MyHostingUnit
HypervisorConnection : MyConnection
RootPath             : /
RootId               :
NetworkPath          : /Network 0.network
NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
Storage              : {/Local storage.storage, /newStorage.storage}
PersonalDiskStorage : {/Local storage.storage}
VMTaggingEnabled     : True
Metadata             : {}
The command adds a new storage location called "newStorage.storage" to the current directory. The dot (.) represents the current location (not its contents).

-------------------------- EXAMPLE 3 --------------------------
XDHyp:HostingUnits\MyHostingUnit\dir *.storage | Add-HypHostingUnitStorage -LiteralPath XDHyp:HostingUnits\MyHostingUnit\newStorage.storage' -StorageType PersonalDiskStorage
HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
HostingUnitName      : MyHostingUnit
HypervisorConnection : MyConnection
RootPath             : /
RootId               :
NetworkPath          : /Network 0.network
NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
Storage              : {/Local storage.storage, /newStorage.storage}
The command adds all of the storage that is available in the hosting unit to the specified hosting unit.

-------------------------- EXAMPLE 4 --------------------------
c:\PS> Add-HypHostingUnitStorage -LiteralPath XDHyp:HostingUnits\MyHostingUnit\StoragePath 'XDHyp:HostingUnits\MyHostingUnits\newStorage.storage' -StorageType PersonalDiskStorage
HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
HostingUnitName      : MyHostingUnit
HypervisorConnection : MyConnection
RootPath             : /
RootId               :
NetworkPath          : /Network 0.network
NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
Storage              : {/Local storage.storage}
The command adds a new storage location called "newStorage.storage" to the hosting unit called "MyHostingUnit".
Add-HypHypervisorConnectionAddress

Oct 10, 2013
Add a connection address to a hypervisor connection.

Syntax

Add-HypHypervisorConnectionAddress [-LiteralPath] <String> [-HypervisorAddress] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

This provides the ability to add addresses by which a hypervisor can be contacted. All the addresses added to a single hypervisor connection are assumed to be equivalent (i.e. they all result in the ability to communicate with the same hypervisors). The hypervisor that the addresses reference is stored at the point of creation of the hypervisor connection. Once this is done all addresses must resolve to this hypervisor to be valid.

The addresses required are: XenServer - The address of the XenServer machines (must all reference the same XenServer pool). VMWare vSphere/ESX - The address of a vCenter server. Microsoft SCVMM/Hyper-V - the address of an SCVMM server.

Related topics

Remove-HypHypervisorConnectionAddress
Get-HypXenServerAddress

Parameters

-LiteralPath <String>
Specifies the path within a Host Service provider to the hypervisor connection item to which to add the address. The path specified must be in one of the following formats; <drive>\Connections\<HypervisorConnectionName> or <drive>\Connections\{HypervisorConnection Uid}\n
| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-HypervisorAddress <String>
Specifies the address to be used. The address will be validated and the hypervisor must be contactable at the address supplied.

XenServer (ConnectionType = XenServer) The address being added must reference the same XenServer pool referenced by any existing addresses for the same connection. VMWare vSphere\ESX (ConnectionType = vCenter) The address being added must be to the same VMWare vCenter as the existing addresses. SCVMM\Hyper-V (ConnectionType = SCVMM) The address being added must be to the same SCVMM server as the existing addresses. Custom Connection Types ?Validation here I assume none need to check?

#PUBS NOTE: ASSUME THIS TO BE UPDATED?#

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |
Specify the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

| Required? | false |
| Default Value | LocalHost. Once a value is provided by any cmdlet, this value will become the default. |
| Accept Pipeline Input? | false |

Return Values

`Citrix.Host.Sdk.HypervisorConnection`

Add-HypHypervisorConnectionAddress returns an object containing the new definition of the hypervisor connection.

`HypervisorConnectionUid <Guid>`

Specifies the unique identifier for the hypervisor connection.

`HypervisorConnectionName <string>`

Specifies the name of the hypervisor connection.

`ConnectionType <Citrix.XDInterServiceTypes.ConnectionType>`

Specifies the connection type of the connection.

- **XenServer** - A Citrix XenServer hypervisor
- **SCVMM** - A Microsoft SCVMM/Hyper-V hypervisor
- **vCenter** - A VMWare vSphere/ESX hypervisor
- **Custom** - A 3rd party hypervisor

`HypervisorAddress <string[]>`

A list of addresses that can be used to communicate with the hypervisor.

`UserName <string>`

The user name that is used when connecting to the hypervisor.
Persistent <Boolean>
Indicates whether the connection is stored in the database or is only a temporary connection that has the same lifetime as the current Powershell session.

PluginId <string>
The Citrix identifier for the Citrix Machine Management plug-in.

Revision <Guid>
Identifier for the current version of the hypervisor connection. This value changes every time a property of the hypervisor connection is updated.

SupportsPvsVMs <Boolean>
Indicates whether the connection can be used as part of a hosting unit and therefore used by the Citrix XenDesktop Machine Creation Services to create virtual machines. Only the built-in supported hypervisor connection types can be used for this (i.e. XenServer, SCVMM and vCenter).

Metadata <Citrix.Host.Sdk.Metadata[]>
A list of key value pairs that can store additional information relating to the hosting unit.

Notes
The address format must be valid for the hypervisor connection. The rules that are applied are as below:

XenServer (HypervisorConnection Type = XenServer)
http:\<IP Address>
or http:\<server Name>
or https:\<server Name>

Note: To use multiple addresses to the same XenServer pool in order to provide failover functionality, all the XenServers in the pool must have a shared block storage device. If using https connections and failover is required, the certificates on the servers must be trusted by all of the controllers (typically this means having a root certificate installed).

VMWare vSphere/ESX (HypervisorConnection Type = vCenter)
http:\<IP Address>\<sdk path>
http:\<server Name>\<sdk path>
or https:\<Server Name>\<sdk path>

Notes:
* http requires manual vCenter configuration; https is the only supported default format.
* Default sdk path is ‘sdk’

Microsoft SCVMM/Hyper-v (HypervisorConnection Type = SCVMM)
IP Address
server Name (short or FQDN)

Note: If XenServers are moved to new XenServer pools after being added to a hypervisor connection, unpredictable results will occur.
Once a XenServer is assigned to a hypervisor connection it should not be moved to a new XenServer pool.

In the case of failure the following errors can be produced.

Error Codes
-----------

**InputConnectionsPathInvalid**
The path provided is not to an item in a sub directory of a hosting unit item.

**HypervisorConnectionAddressForeignKeyObjectDoesNotExist**
The hypervisor connection to which the address is to be added could not be located.

**ConnectionAddressInvalid**
The address could not be used to contact a hypervisor or the validation rules have not been met.

**HypervisorConnectionAddressDuplicateObjectExists**
The address specified is already part of the hypervisor connection.

**HypervisorInMaintenanceMode**
The hypervisor for the connection is in maintenance mode.

**DatabaseError**
An error occurred in the service whilst attempting a database operation.

**DatabaseNotConfigured**
The operation could not be completed as the database for the service is not configured.

**DataStoreException**
An error occurred in the service whilst attempting a database operation - communication to database failed for various reasons.

**CommunicationError**
An error occurred whilst communicating with the service.

**PermissionDenied**
The user does not have administrative rights to perform this operation.

**ExceptionThrown**
An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples
--------

**EXAMPLE 1**

c:\PS> Add-HypHypervisorConnectionAddress -LiteralPath XDHyp:\Connections\MyConnection -HypervisorAddress http\:myserver.com
Add the address 'http:\myserver.com' to the hypervisor connection called "MyConnection".
Add-HypHypervisorConnectionMetadata

Oct 10, 2013
Adds metadata on the given HypervisorConnection.

Syntax

Add-HypHypervisorConnectionMetadata [-HypervisorConnectionUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Add-HypHypervisorConnectionMetadata [-HypervisorConnectionUid] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Add-HypHypervisorConnectionMetadata [-HypervisorConnectionName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Add-HypHypervisorConnectionMetadata [-HypervisorConnectionName] <String> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Add-HypHypervisorConnectionMetadata [-InputObject] <HypervisorConnection[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Add-HypHypervisorConnectionMetadata [-InputObject] <HypervisorConnection[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Provides the ability for additional custom data to be stored against given HypervisorConnection objects. This cmdlet will not overwrite existing metadata on an object - use the Set-HypHypervisorConnectionMetadata cmdlet instead.

Related topics

Set-HypHypervisorConnectionMetadata
Remove-HypHypervisorConnectionMetadata

Parameters

-HypervisorConnectionUid <Guid>
Id of the HypervisorConnection

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-HypervisorConnectionName <String>
Name of the HypervisorConnection

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-InputObject <HypervisorConnection[]>
Objects to which the metadata is to be added.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |
### -Map<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with new-object `"System.Collections.Generic.Dictionary[String,String]"`).

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

### -Name<String>

Specifies the property name of the metadata to be added. The property must be unique for the HypervisorConnection specified. The property cannot contain any of the following characters \/:.*?=<>|[]()

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

### -Value<String>

Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

### -LoggingId<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

### Return Values
Citrix.Host.Sdk.Metadata

Add-HypHypervisorConnectionMetadata returns an array of objects containing the new definition of the metadata.

Property <string>
   Specifies the name of the property.

Value <string>
   Specifies the value for the property.

Notes
If the command fails, the following errors can be returned.

Error Codes
----------

InvalidParameterCombination
   The cmdlet parameters are inconsistent.

UnknownObject
   One of the specified objects was not found.

DuplicateObject
   One of the specified metadata already exists.

DatabaseError
   An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
   The operation could not be completed because the database for the service is not configured.

DataStoreException
   An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
   You do not have permission to execute this command.

AuthorizationError
   There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
   The operation could not be performed because of a configuration logging error.

CommunicationError
   There was a problem communicating with the remote service.

ExceptionThrown
   An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS> Add-HypHypervisorConnectionMetadata -HypervisorConnectionUid 4CECC26E-48E1-423F-A1F0-2A06DD0D0805C -Name property -Value value
<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the HypervisorConnection with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Add-HypHypervisorConnectionScope

Oct 10, 2013
Add the specified HypervisorConnection(s) to the given scope(s).

Syntax
Add-HypHypervisorConnectionScope [-Scope] <String[]> -InputObject <HypervisorConnection[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]
Add-HypHypervisorConnectionScope [-Scope] <String[]> -HypervisorConnectionUid <Guid[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]
Add-HypHypervisorConnectionScope [-Scope] <String[]> -HypervisorConnectionName <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description
The AddHypHypervisorConnectionScope cmdlet is used to associate one or more HypervisorConnection objects with given scope(s).

There are multiple parameter sets for this cmdlet, allowing you to identify the HypervisorConnection objects in different ways:
- HypervisorConnection objects can be piped in or specified by the InputObject parameter
- The HypervisorConnectionUid parameter specifies objects by HypervisorConnectionUid
- The HypervisorConnectionName parameter specifies objects by HypervisorConnectionName (supports wildcards)

To add a HypervisorConnection to a scope you need permission to change the scopes of the HypervisorConnection and permission to add objects to all of the scopes you have specified.

If the HypervisorConnection is already in a scope, that scope will be silently ignored.

Related topics
Remove-HypHypervisorConnectionScope
Get-HypScopedObject

Parameters
-Scope <String[]>
Specifies the scopes to add the objects to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>True</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

-InputObject <HypervisorConnection[]>
 Specifies the HypervisorConnection objects to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>True</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>
## Accept Pipeline Input?

**true (ByValue, ByPropertyName)**

## HypervisorConnectionUid

**<Guid[]>**

Specifies the HypervisorConnection objects to be added by HypervisorConnectionUid.

- **Required?** true
- **Default Value**
- **Accept Pipeline Input?** true (ByValue, ByPropertyName)

## HypervisorConnectionName

**<String[]>**

Specifies the HypervisorConnection objects to be added by HypervisorConnectionName.

- **Required?** true
- **Default Value**
- **Accept Pipeline Input?** true (ByValue, ByPropertyName)

## LoggingId

**<Guid>**

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

- **Required?** false
- **Default Value**
- **Accept Pipeline Input?** false

## AdminAddress

**<String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

- **Required?** false
- **Default Value** Localhost. Once a value is provided by any cmdlet, this value becomes the default.
- **Accept Pipeline Input?** false

### Return Values

None

### Notes

If the command fails, the following errors can be returned.

### Error Codes

---------

[https://docs.citrix.com](https://docs.citrix.com)  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.2065
UnknownObject
One of the specified objects was not found.

ScopeNotFound
One of the specified scopes was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

---------------------------- EXAMPLE 1 ----------------------------
c:\PS> Add-HypHypervisorConnectionScope Finance -HypervisorConnectionUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
Adds a single HypervisorConnection to the 'Finance' scope.

---------------------------- EXAMPLE 2 ----------------------------
c:\PS> Add-HypHypervisorConnectionScope Finance,Marketing -HypervisorConnectionUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
Adds a single HypervisorConnection to the multiple scopes.

---------------------------- EXAMPLE 3 ----------------------------
c:\PS> Get-HypHypervisorConnection | Add-HypHypervisorConnectionScope Finance
Adds all visible HypervisorConnection objects to the 'Finance' scope.

---------------------------- EXAMPLE 4 ----------------------------
c:\PS> Add-HypHypervisorConnectionScope Finance -HypervisorConnectionName A*
Adds HypervisorConnection objects with a name starting with an 'A' to the 'Finance' scope.
Add-HypMetadata

Oct 10, 2013
Adds metadata to a hypervisor connection or a hosting unit.

Syntax

Detailed Description
Provides the ability for additional custom data to be stored against a hosting unit or hypervisor Connection. This data is never used by the Machine Creation Services, and is provided for consumers of the services to store any data that might be required for their operations. The metadata is returned along with the hypervisor connection or hosting unit that it is assigned to.

Related topics
Remove-HypMetadata

Parameters
-LiteralPath <String>
Specifies the path within a hosting unit provider to the hosting unit or hypervisor connection item to which to add the metadata. The path specified must be in one of the following formats: <drive>:\HostingUnits\HostingUnitName> or <drive>:\HostingUnits\{<HostingUnit Uid> or <drive>:\Connections\Connection Name> or <drive>:\Connections\{<Connection Uid>

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-Property <String>
Specifies the property name of the metadata to be added. The property must be unique for the item specified by the path. The property cannot contain any of the following characters \\/:.*?=<> | []"

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-Value <String>
Specifies the value for the property.
- **LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Input Type**

System.String You can pipe a string the contains a path to Add-HypMetadata (Path parameter).

**Return Values**

Citrix.Host.Sdk.Metadata

Add-HypMetadata returns an object containing the new definition of the metadata.

**Property**<string>

Specifies the property of the metadata.

**Value**<string>

Specifies the value of the metadata.

**Notes**

In the case of failure the following errors can be produced.
Error Codes
----------

InvalidPath
The path provided is not in the required format.

HostingUnitMetadataForeignKeyObjectDoesNotExist
The hosting unit supplied in the path does not exist.

HypervisorConnectionMetadataForeignKeyObjectDoesNotExist
The hypervisor connection supplied in the path does not exist.

HostingUnitMetadataDuplicateObjectExists
Metadata for the specified hosting unit item already exists with the same property name.

HypervisorConnectionMetadataDuplicateObjectExists
Metadata for the specified hypervisor connection item already exists with the same property name.

MetadataContainerUndefined
The specified path does not reference a hosting unit or a hypervisor connection.

DatabaseError
An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured
The operation could not be completed as the database for the service is not configured.

DataStoreException
An error occurred in the service whilst attempting a database operation - communication to database failed for various reasons.

CommunicationError
An error occurred whilst communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples
--- EXAMPLE 1 ---

c:\PS>Add-HypMetadata -LiteralPath XDHyp:\Connections\MyConnection -Property MyProperty -Value MyValue

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>MyProperty</td>
<td>MyValue</td>
</tr>
</tbody>
</table>

The command adds the metadata with the property name of "MyProperty" and value of "MyValue" to the hypervisor connection item called "MyConnection".

--- EXAMPLE 2 ---

c:\PS>dir xdhyp\connections\Citrix* | Add-HypMetadata -Property MyProperty -Value MyValue

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>MyProperty</td>
<td>MyValue</td>
</tr>
<tr>
<td>MyProperty</td>
<td>MyValue</td>
</tr>
<tr>
<td>MyProperty</td>
<td>MyValue</td>
</tr>
</tbody>
</table>

The command adds the metadata with the property name of "MyProperty" and value of "MyValue" to all the hypervisor connection items that begin with the string "Citrix".
Get-HypConfigurationDataForItem

Oct 10, 2013
Retrieves the configuration data for an item in the Host Service provider path. Note: For this release, only VM items are supported for this operation.

Syntax
Get-HypConfigurationDataForItem [-LiteralPath] <String> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
This command provides a mechanism for retrieving extra data about an entry in the hosting unit service provider. The referenced item must be contained within the connections directory in the provider (i.e. XDHyp:Connections).

This mechanism is used for obtaining data that is not required frequently and/or has a high overhead associated with its retrieval (so as to maintain the responsiveness of the provider). For this release of the PowerShell snap-in the only provider items that can be used with this operation are VM items. For a VM this provides a mechanism to obtain the number of CPUs, the amount of Memory (in MB) and hard disk drive capacity (GB).

Related topics
Get-Item

Parameters
-LiteralPath <String>
Specifies the path within a hosting unit provider to the item for which configuration data is to be retrieved. The path specified must be in one of the following formats; <drive>\Connections\<Connection Name>\<Item Path of VM object> or <drive>\Connections\{<connection Uid>\<Item Path of VM object>} or <drive>\HostingUnits\<HostingUnit Name>\<Item Path of VM object> or <drive>\HostingUnits\{<hostingUnit Uid>\<Item Path of VM object>

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>localhost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Input Type
System.string You can pipe a string the contains a path to Get-HypConfigurationDataForItem

Return Values
PSObject
Get-HypConfigurationDataForItem returns a PSObject containing the properties that are appropriate for the item specified by the path.

Properties for VM Item

----------------------
CPUCount <int>
Specifies the number of CPUs assigned to the VM.

MemoryMB <int>
The amount of memory allocated to the VM.

HardDiskSizeGB <int>
The capacity of the primary hard drive assigned to the VM.

Notes
For this release this cmdlet only provides configuration data for VM objects in the provider. Using a path to an item that is not a VM will result in a error.

In the case of failure the following errors can be produced.

Error Codes

----------
InputHypervisorItemPathInvalid
The path provided is not to an item in a sub directory of a connection item or a hosting unit item.

InvalidHypervisorItemPath
No item exists with the specified path.

InvalidHypervisorItem
The item specified by the path exists, but is not a VM Item.

DatabaseError
An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured
The operation could not be completed as the database for the service is not configured.

DataStoreException

An error occurred in the service whilst attempting a database operation - communication to database failed for various reasons.

CommunicationError

An error occurred whilst communicating with the service.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

HypervisorPermissionDenied

The hypervisor login used does not provide authorization to access the data for this item.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Get-HypConfigurationDataForItem -LiteralPath XDHyp:\Connections\MyConnection\MyVm.vm

CpuCount       MemoryMB    HardDiskSizeGB
-------------   ----------    --------------
     1           1024          24

This command gets the configuration properties for a VM called 'MyVm.vm' within a hypervisor connection called 'MyConnection'.

-------------------------- EXAMPLE 2 --------------------------

XDHyp:\HostingUnits\PS>Get-HypConfigurationDataForItem -LiteralPath .\MyVm.vm

CpuCount       MemoryMB    HardDiskSizeGB
-------------   ----------    --------------
     1           1024          24

This command gets the configuration properties for a VM called 'MyVm.vm' within the current directory. The dot (.) represents the current location (not its contents).

-------------------------- EXAMPLE 3 --------------------------

C:\PS>(Get-HypConfigurationDataForItem -LiteralPath XDHyp:\Connections\MyConnection\MyVm.vm).CPUCount

This command gets the CPU count for a VM called 'MyVm.vm'. The CPUCount is just one property of the VM items. To see all properties of an item, type ":(Get-HypConfigurationDataForItem &lt;ItemPath&gt;) | Get-Member".
Get-HypConfigurationObjectForItem

Oct 10, 2013

Retrieves the configuration data for an item in the Host Service provider path. Note: For this release, only VM items are supported for this operation.

Syntax

Get-HypConfigurationObjectForItem [-LiteralPath] <String> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

This command provides a mechanism for retrieving extra data about an entry in the hosting unit service provider. The referenced item must be contained within the connections directory in the provider (i.e. XDHyp:\Connections).

This mechanism is used for obtaining data that is not required frequently and/or has a high overhead associated with its retrieval (so as to maintain the responsiveness of the provider). For this release of the PowerShell snap-in the only provider items that can be used with this operation are VM items. For a VM this provides a mechanism to obtain the number of CPUs, the amount of Memory (in MB) and hard disk drive capacity (GB).

Related topics

Get-Item

Parameters

-LiteralPath <String>

Specifies the path within a hosting unit provider to the item for which configuration data is to be retrieved. The path specified must be in one of the following formats; <drive>\Connections\<Connection Name>\<Item Path of VM object> or <drive>\Connections\{<connection Uid>\<Item Path of VM object>\} or <drive>\HostingUnits\<HostingUnit Name>\<Item Path of VM object> or <drive>\HostingUnits\{<hostingUnit Uid>\<Item Path of VM object>\}

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Input Type
System.String
You can pipe a string that contains a path to Get-HypConfigurationDataForItem.

Return Values
PSObject
Get-HypConfigurationDataForItem returns a PSObject containing the properties that are appropriate for the item specified by the path.

Properties for VM Item

----------------------
CPUCount <int>
Specifies the number of CPUs assigned to the VM.

MemoryMB <int>
The amount of memory allocated to the VM.

HardDiskSizeGB <int>
The capacity of the primary hard drive assigned to the VM.

Network Map
The networks that this VM or Snapshot is connected to.

Notes
For this release this cmdlet only provides configuration data for VM objects in the provider. Using a path to an item that is not a VM will result in an error.

In the case of failure the following errors can be produced.

Error Codes

----------

InputHypervisorItemPathInvalid
The path provided is not to an item in a sub directory of a connection item or a hosting unit item.

InvalidHypervisorItemPath
No item exists with the specified path.

InvalidHypervisorItem
The item specified by the path exists, but is not a VM Item.

DatabaseError


An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured

The operation could not be completed as the database for the service is not configured.

DataStoreException

An error occurred in the service whilst attempting a database operation - communication to database failed for for various reasons.

CommunicationError

An error occurred whilst communicating with the service.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

HypervisorPermissionDenied

The hypervisor login used does not provide authorization to access the data for this item.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Get-HypConfigurationDataForItem -LiteralPath XDHyp:\Connections\MyConnection\MyVm.vm

<table>
<thead>
<tr>
<th>CpuCount</th>
<th>MemoryMB</th>
<th>HardDiskSizeGB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1024</td>
<td>24</td>
</tr>
</tbody>
</table>

This command gets the configuration properties for a VM called 'MyVm.vm' within a hypervisor connection called 'MyConnection'.

-------------------------- EXAMPLE 2 --------------------------

XDHyp:\HostingUnits\PS>Get-HypConfigurationDataForItem -LiteralPath .\MyVm.vm

<table>
<thead>
<tr>
<th>CpuCount</th>
<th>MemoryMB</th>
<th>HardDiskSizeGB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1024</td>
<td>24</td>
</tr>
</tbody>
</table>

This command gets the configuration properties for a VM called 'MyVm.vm' within the current directory. The dot (.) represents the current location (not its contents).

-------------------------- EXAMPLE 3 --------------------------

C:\PS>(Get-HypConfigurationDataForItem -LiteralPath XDHyp:\Connections\MyConnection\MyVm.vm).CPUCount
This command gets the CPU count for a VM called 'MyVm.vm'. The CPUCount is just one property of the VM items. To see all properties of an item, type "(Get-HypConfigurationDataForItem <ItemPath>) | Get-Member".
Get-HypDBConnection

Oct 10, 2013
Gets the database string for the specified data store used by the Host Service.

Syntax
Get-HypDBConnection [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Returns the database connection string for the specified data store.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

Related topics
Get-HypServiceStatus
Set-HypDBConnection
Test-HypDBConnection

Parameters
-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Return Values
system: string
The database connection string configured for the Host Service.

Notes
If the command fails, the following errors can be returned.

Error Codes
----------------
NoDBConnections
The database connection string for the Host Service has not been specified.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-HypDBConnection

Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True
Get the database connection string for the Host Service.
Get-HypDBSchema

Oct 10, 2013
Gets a script that creates the Host Service database schema for the specified data store.

Syntax


Detailed Description

Gets SQL scripts that can be used to create a new Host Service database schema, add a new Host Service to an existing site, remove a Host Service from a site, or create a database server logon for a Host Service. If no Sid parameter is provided, the scripts obtained relate to the currently selected Host Service instance, otherwise the scripts relate to Host Service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied. The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to a Host SDK cmdlet. The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured. The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SMDCMD mode'. The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to Host Service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to Host Service roles

If ScriptType is Evict, the returned script contains:

- Removal of Host Service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

If the service uses two data stores they can exist in the same database. You do not need to configure a database before using this command.
Related topics

Set-HypDBConnection

Parameters

- **DatabaseName**<String>

  Specifies the name of the database for which the schema will be generated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ServiceGroupName**<String>

  Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the Host services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ScriptType**<ScriptTypes>

  Specifies the type of database script returned. Available script types are:

  **Database**

  Returns a full database script that can be used to create a database schema for the Host Service in a database instance that does not already contain a schema for this service. The DatabaseName and ServiceGroupName parameters must be specified to create a script of this type.

  **Instance**

  Returns a permissions script that can be used to add further Host services to an existing database instance that already contains the full Host service schema, associating the services to the Service Group. The Sid parameter can optionally be specified to create a script of this type.

  **Login**

  Returns a database logon script that can be used to add the required logon accounts to an existing database instance that contains the Host Service schema. This is used primarily when creating a mirrored database environment. The DatabaseName parameter must be specified to create a script of this type.

  **Evict**
Returns a script that can be used to remove the specified Host Service from the database entirely. The DatabaseName and Sid parameters must be specified to create a script of this type.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Database</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LocalDatabase**<SwitchParameter>

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for Host services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Sid**<String>

Specifies the SID of the controller on which the Host Service instance to remove from the database is running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

**System.String**

A string containing the required SQL script for application to a database.

Notes
The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

Create the database schema.
Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned.

Error Codes
-----------
GetSchemasFailed
The database schema could not be found.
ActiveDirectoryAccountResolutionFailed
The specified Active Directory account or Group could not be found.
DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.
PermissionDenied
You do not have permission to execute this command.
AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-HypDBSchema -DatabaseName MyDB -ServiceGroupName  MyServiceGroup > c:\HypSchema.sql
Get the full database schema for site data store of the Host Service and copy it to a file called 'c:\HypSchema.sql'.
This script can then be used to create the schema in a pre-existing database named 'MyDB' that does not already contain a Host Service site schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-HypDBSchema -DatabaseName MyDB -scriptType Login > c:\HostLogins.sql
Get the logon scripts for the Host Service.
Get-HypDBVersionChangeScript

Oct 10, 2013
Gets a script that updates the Host Service database schema.

Syntax

Get-HypDBVersionChangeScript -DatabaseName <String> -TargetVersion <Version> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns a database script that can be used to upgrade or downgrade the site or secondary schema for the Host Service from the current schema version to a different version.

Related topics

Get-HypInstalledDBVersion

Parameters

-DatabaseName <String>

Specifies the name of the database instance to which the update applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-TargetVersion <Version>

Specifies the version of the database you want to update to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values

System.Management.Automation.PSObject

A PSObject containing the required SQL script for application to a database.

Notes

The PSObject returned by this cmdlet contains the following properties:

-- Script The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

-- NeedExclusiveAccess Indicates whether all services in the service group must be shut down during the update or not.

-- CanUndo Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any Host services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-HypServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned.

Error Codes

---------

NoOp

The operation was successful but had no effect.

NoDBCConnections

The database connection string for the Host Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $update = Get-HypDBVersionChangeScript -DatabaseName MyDb -TargetVersion 1.0.75.0
C:\PS> $update.Script > update_75.sql

Gets an SQL update script to update the current schema to version 1.0.75.0. The resulting update_75.sql script is suitable for direct use with the SQL Server SQLCMD utility.
Get-HypHypervisorPlugin

Oct 10, 2013
Gets the available hypervisor types.

Syntax

Get-HypHypervisorPlugin [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The provides the ability to retrieve a list of all the available hypervisor types, and their localized names.

Related topics

New-Item

Parameters

- **<AdminAddress><String>**

  Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.Host.Sdk.HypervisorPlugin

Get-HypHypervisorPlugin returns a list of objects containing the definition of the hypervisor plug-ins.

ConnectionType <Citrix.XDInterServiceTypes.ConnectionType>

The hypervisor connection type. This can be one of the following:

- XenServer - Xenserver hypervisor
- SCVMM - Microsoft SCVMM/Hyper-v
- vCenter - VMware vSphere/ESX
- Custom - a third party hypervisor

DisplayName <string>

The localized display name (localized using the locale of the Powershell snap-in session)

PluginFactoryName <string>
The name of the hypervisor plug-in factory used to manage the hypervisor connections.

Notes

To utilize third party plug-ins, the plug-in assemblies must be installed into the appropriate location on each controller machine that forms part of the Citrix controller site. Failure to do this will result in unpredictable behavior, especially during service failover conditions.

In the case of failure the following errors can be produced.

Error Codes

----------

DatabaseError
An error occurred in the service whilst attempting a database operation.

CommunicationError
An error occurred whilst communicating with the service.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown
An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS> Get-HypHypervisorPlugin | Format-Table -AutoSize

<table>
<thead>
<tr>
<th>ConnectionType</th>
<th>DisplayName</th>
<th>PluginFactoryName</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCVMM Microsoft virtualization</td>
<td>MicrosoftPSFactory</td>
<td></td>
</tr>
<tr>
<td>VCenter VMware virtualization</td>
<td>VmwareFactory</td>
<td></td>
</tr>
<tr>
<td>XenServer Citrix XenServer</td>
<td>XenFactory</td>
<td></td>
</tr>
</tbody>
</table>

Get the available hypervisor management plug-ins.
Get-HypInstalledDBVersion

Oct 10, 2013
Gets a list of all available database schema versions for the Host Service.

Syntax

Get-HypInstalledDBVersion [-Upgrade] [-Downgrade] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns the current version of the Host Service database schema, if no flags are set, otherwise returns versions for which upgrade or downgrade scripts are available and have been stored in the database.

Related topics

Parameters

-**Upgrade**<SwitchParameter>

Specifies that only schema versions to which the current database version can be updated should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Downgrade**<SwitchParameter>

Specifies that only schema versions to which the current database version can be reverted should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
System.Version

The Get-HypInstalledDbVersion command returns objects containing the new definition of the Host Service database schema version.

Major <Integer>
Minor <Integer>
Build <Integer>
Revision <Integer>

Notes
If the command fails, the following errors can be returned.

Error Codes
-----------

InvalidParameterCombination
Both the Upgrade and Downgrade flags were specified.

NoOp
The operation was successful but had no effect.

NoDBConnections
The database connection string for the Host Service has not been specified.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-HypInstalledDBVersion

Major  Minor  Build  Revision
-----  -----  -----  --------
5      6      0      0

Get the currently installed version of the Host Service database schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-HypInstalledDBVersion -Upgrade

Major  Minor  Build  Revision
-----  -----  -----  --------
6      0      0      0

Get the versions of the Host Service database schema for which upgrade scripts are supplied.
Get-HypScopedObject

Oct 10, 2013
Gets the details of the scoped objects for the Host Service.

Syntax


Detailed Description

Returns a list of directly scoped objects including the names and identifiers of both the scope and object as well as the object description for display purposes.

There will be at least one result for every directly scoped object. When an object is associated with multiple scopes the output contains one result per scope duplicating the object details.

No records are returned for the All scope, though if an object is not in any scope a result with a null ScopeId and ScopeName will be returned.

Related topics

Parameters

- **ScopeId**<Guid>

  Gets scoped object entries for the given scope identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

  Accept Pipeline Input? | true (ByPropertyName)

- **ScopeName**<String>

  Gets scoped object entries with the given scope name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

  Accept Pipeline Input? | true (ByPropertyName)

- **ObjectType**<ScopedObjectType>

  Gets scoped object entries for objects of the given type.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

  Accept Pipeline Input? | true (ByPropertyName)
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ObjectId</td>
<td>&lt;String&gt;</td>
<td>false</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
<tr>
<td>-ObjectName</td>
<td>&lt;String&gt;</td>
<td>false</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
<tr>
<td>-Description</td>
<td>&lt;String&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>-Property</td>
<td>&lt;String[]&gt;</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-ReturnTotalRecordCount</td>
<td>&lt;SwitchParameter&gt;</td>
<td></td>
<td>When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Hyp_Filtering for details.</td>
<td></td>
</tr>
</tbody>
</table>
**Required?** | false  
---|---  
**Default Value** | False  
**Accept Pipeline Input?** | false

**-MaxRecordCount</Int32>**

Specifies the maximum number of records to return.

| Required? | false  
|---|---  
| Default Value | 250  
| Accept Pipeline Input? | false

**-Skip</Int32>**

Skips the specified number of records before returning results. Also reduces the count returned by `-ReturnTotalRecordCount`.

| Required? | false  
|---|---  
| Default Value | 0  
| Accept Pipeline Input? | false

**-SortBy</String>**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false  
|---|---  
| Default Value | The default sort order is by name or unique identifier.  
| Accept Pipeline Input? | false

**-Filter</String>**

Gets records that match a PowerShell-style filter expression. See about_Hyp_Filtering for details.

| Required? | false  
|---|---  
| Default Value  
| Accept Pipeline Input? | false
**-AdminAddress<br>**

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**

Citrix.Host.Sdk.ScopedObject

The Get-HypScopedObject command returns an object containing the following properties:

**Scopeld <Guid?>**

Specifies the unique identifier of the scope.

**ScopeName <String>**

Specifies the display name of the scope.

**ObjectType <ScopedObjectType>**

Type of the object this entry relates to.

**ObjectId <String>**

Unique identifier of the object.

**ObjectName <String>**

Display name of the object

**Description <String>**

Description of the object (possibly $null if the object type does not have a description).

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

----------

UnknownObject

One of the specified objects was not found.

PartialData
Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-HypScopedObject -ObjectType Scheme

Scopeld     : eff6f464-f1ee-4442-add3-99982e0cec01
ScopeName   : Sales
ObjectType  : Scheme
ObjectId    : cd4174ee-9e4b-4e57-b126-9dbf757fe493
ObjectName  : MyExampleScheme
Description : Test scheme

Scopeld     : 304e0fa7-d390-47f0-a94f-7e956a324c41
ScopeName   : Finance
Gets all of the scoped objects with type Scheme. The example output shows a scheme object (MyExampleScheme) in two scopes Sales and Finance, and another scheme (AnotherScheme) that is not in any scope. The ScopeId and ScopeName values returned are null in the final record.
Get-HypService

Oct 10, 2013
Gets the service record entries for the Host Service.

Syntax

Detailed Description
Returns instances of the Host Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Related topics
Parameters
-Property <String[]>
Specifies the properties to be be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-ReturnTotalRecordCount <SwitchParameter>
When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Hyp_Filtering for details.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

-MaxRecordCount <Int32>
Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |
-Skip<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-SortBy<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Filter<String>

Gets records that match a PowerShell-style filter expression. See about_Hyp_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
Citrix.Host.Sdk.Service
The Get-HypServiceInstance command returns an object containing the following properties.

Uid <Integer>
Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>
Specifies the unique identifier for the service instance.

DNSName <String>
Specifies the domain name of the host on which the service runs.

MachineName <String>
Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>
Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>
Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>
Specifies the date and time at which the service was last stopped or restarted.

OSType
Specifies the operating system installed on the host on which the service runs.

OSVersion
Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion
Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>
Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>
Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>
Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.
Notes
If the command fails, the following errors can be returned.

Error Codes
----------

UnknownObject
One of the specified objects was not found.

PartialData
Only a subset of the available data was returned.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase
The query required to get the database was not defined.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------- EXAMPLE 1 --------------------------
c:\PS>Get-HypService

Uid : 1
ServiceHostId : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName : MyServer.company.com
MachineName : MYSERVER
CurrentState : On
LastStartTime : 04/04/2011 15:25:38
LastActivityTime : 04/04/2011 15:33:39
OSType : Win32NT
OSVersion : 6.1.7600.0
ServiceVersion : 5.1.0.0
DatabaseUserName : NT AUTHORITY\NETWORK SERVICE
SID : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {MySiteService1, MySiteService2...}

Get all the instances of the Host Service running in the current service group.
Get-HypServiceAddedCapability

Oct 10, 2013
Gets any added capabilities for the Host Service on the controller.

Syntax
Get-HypServiceAddedCapability [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Enables updates to the Host Service on the controller to be detected.
You do not need to configure a database connection before using this command.

Related topics
Parameters

- **-AdminAddress**<String>
  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.String
List containing added capabilities.

String containing added capabilities.

Notes
If the command fails, the following errors can be returned.

Error Codes

--------------
DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-HypServiceAddedCapability
Get the added capabilities of the Host Service.
Get-HypServiceInstance

Oct 10, 2013
Gets the service instance entries for the Host Service.

Syntax
Get-HypServiceInstance [-AdminAddress <String>] [CommonParameters]

Detailed Description
Returns service interfaces published by the instance of the Host Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Related topics
Parameters
-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Return Values
Citrix.Host.Sdk.ServiceInstance
The Get-HypServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

\nSpecifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

\nSpecifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

\nSpecifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.
ServiceType <String>

- Specifies the service instance type. For this service, the service instance type is always Hyp.

Address

- Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

- Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

- Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

- Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

- Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

- Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:
  - SDK - for PowerShell operations
  - InterService - for operations between different services
  - Peer - for communications between services of the same type

Metadata <Citrix.Host.Sdk.Metadata[]>

- The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

- If the command fails, the following errors can be returned.

Error Codes

------------

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-HypServiceInstance

Address     : http://MyServer.com:80/Citrix/SdkHostingUnitService
Binding     : wcf_HTTP_kerb
InterfaceType: SDK
Metadata     :
MetadataMap  :
ServiceAccount: ENG\MyAccount$
ServiceAccountSid: S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName : MyServiceGroup
ServiceGroupUid  : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceUid: 00000000-0000-0000-0000-000000000000
ServiceType    : Hyp
Version       : 1

Address     : http://MyServer.com:80/Citrix/SdkHostingUnitService/IServiceApi
Binding     : wcf_HTTP_kerb
InterfaceType: InterService
Metadata     :
MetadataMap  :
ServiceAccount: ENG\MyAccount
ServiceAccountSid: S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName : MyServiceGroup
ServiceGroupUid : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceId : 00000000-0000-0000-0000-000000000000
ServiceType : Hyp
Version : 1

Get all instances of the Host Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.
Get-HypServiceStatus

Oct 10, 2013
Gets the current status of the Host Service on the controller.

Syntax

Get-HypServiceStatus [-AdminAddress <String>] [CommonParameters]

Detailed Description

Enables the status of the Host Service on the controller to be monitored. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured.

Related topics

Set-HypDBConnection
Test-HypDBConnection
Get-HypDBConnection
Get-HypDBSchema

Parameters

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

Return Values


The Get-HypServiceStatus command returns an object containing the status of the Host Service together with extra diagnostics information.

DBUnconfigured

The Host Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the Host Service. This may be because the service attempted to log on
with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured
The expected stored procedures are missing from the database. This may be because the Host Service schema has not been added to the database.

DBNotFound
The specified database could not be located with the configured connection string.

DBNewerVersionThanService
The version of the Host Service currently in use is incompatible with the version of the Host Service schema on the database. Upgrade the Host Service to a more recent version.

DBOlderVersionThanService
The version of the Host Service schema on the database is incompatible with the version of the Host Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress
A database schema upgrade is currently in progress.

OK
The Host Service is running and is connected to a database containing a valid schema.

Failed
The Host Service has failed.

Unknown
(0) The service status cannot be determined.

Notes
If the command fails, the following errors can be returned.

Error Codes
-----------

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------- EXAMPLE 1 --------------------

c:\PS>Get-HypServiceStatus

DBUnconfigured
Get the current status of the Host Service.
Get-HypVMMacAddress

Oct 10, 2013
Retrieves a list the MAC addresses for the VMs in the specified connection.

Syntax

Get-HypVMMacAddress [-LiteralPath] <String> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

This command obtains a list of MAC addresses of all the virtual machines in the connection specified.

Related topics

Get-Item

Parameters

-LiteralPath <String>
The path to a connection item in the hosting provider. Paths to anything other than a connection item will result in an error being returned. The path can be provided as either <drive>:\connections\<Connection Name> or <drive>:\connections\ {<Connection Uid>}

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
</tbody>
</table>

Input Type

System.String You can pipe a string the contains a path to Get-HypConfigurationDataForItem

Return Values

Citrix.Host.Sdk.HypervisorVMObject

Get-HypVMMacAddress returns an object containing the following properties.
MacAddress <string> - specifies the MAC address of the VM.

VMId <string> - specifies the identifier for the VM as defined in the hypervisor hosting it.

Notes
The path must refer to a connection item. Hosting unit items are not valid.

In the case of failure the following errors can be produced.

Error Codes

----------

InputConnectionsPathInvalid
If the path is not provided in an expected format an InputConnectionsPathInvalid error will result.

HypervisorConnectionObjectNotFound
The hypervisor connection object specified cannot be found. Why is this different from invalid Path?#AUTHOR'S NOTE - THIS NEEDS UPDATING#

HypervisorInMaintenanceMode
The hypervisor for the connection is in maintenance mode.

DatabaseError
An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured
The operation could not be completed as the database for the service is not configured.

DataStoreException
An error occurred in the service whilst attempting a database operation - communication to database failed for various reasons.

CommunicationError
An error occurred whilst communicating with the service.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown
An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples
<table>
<thead>
<tr>
<th>MacAddress</th>
<th>VMId</th>
</tr>
</thead>
<tbody>
<tr>
<td>52:b0:1c:ed:60:fa</td>
<td>f3395d2a-a196-41c2-e37d-764acf871599</td>
</tr>
<tr>
<td>62:6f:04:0d:5:af</td>
<td>5f4457b0-cc3c-f806-8ca7-5f57e4bf2d1</td>
</tr>
<tr>
<td>4e:a5:9f:00:b2:0c</td>
<td>3115177b-85a9-d8ee-d0f9-0c7437483c09</td>
</tr>
</tbody>
</table>

This command gets the MAC addresses for the connection called "MyConnection".

-------------------------- EXAMPLE 2 --------------------------

<table>
<thead>
<tr>
<th>MacAddress</th>
<th>VMId</th>
</tr>
</thead>
<tbody>
<tr>
<td>52:b0:1c:ed:60:fa</td>
<td>f3395d2a-a196-41c2-e37d-764acf871599</td>
</tr>
<tr>
<td>62:6f:04:0d:5:af</td>
<td>5f4457b0-cc3c-f806-8ca7-5f57e4bf2d1</td>
</tr>
<tr>
<td>4e:a5:9f:00:b2:0c</td>
<td>3115177b-85a9-d8ee-d0f9-0c7437483c09</td>
</tr>
</tbody>
</table>

This command gets the MAC addresses for the connection at the current directory. The dot (.) represents the current location (not its contents).

-------------------------- EXAMPLE 3 --------------------------

<table>
<thead>
<tr>
<th>MacAddress</th>
<th>VMId</th>
</tr>
</thead>
<tbody>
<tr>
<td>52:b0:1c:ed:60:fa</td>
<td>f3395d2a-a196-41c2-e37d-764acf871599</td>
</tr>
<tr>
<td>62:6f:04:0d:5:af</td>
<td>5f4457b0-cc3c-f806-8ca7-5f57e4bf2d1</td>
</tr>
<tr>
<td>4e:a5:9f:00:b2:0c</td>
<td>3115177b-85a9-d8ee-d0f9-0c7437483c09</td>
</tr>
</tbody>
</table>

This command gets the MAC addresses for the connection that has a ConnectionUid of 268c66db-9b8c-47f6-9265-42326dbff006.
Get-HypXenServerAddress

Oct 10, 2013

Gets all the available addresses for a XenServer hypervisor connection.

Syntax

Get-HypXenServerAddress [-LiteralPath] <String> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

This cmdlet provides the ability to retrieve all the available hypervisor connection addresses that can be used to connect to the same XenServer pool. When used in conjunction with Add-HypHypervisorAddress, this can be used to easily populate a connection with all the addresses that can be used to provide full failover support for a XenServer connection.

If the addresses are HTTPS addresses, the command will use the certificates installed on the XenServers to provide suitable HTTPS addresses where possible. Only servers that can be resolved will be returned.

Related topics

Add-HypHypervisorConnectionAddress

Parameters

-LiteralPath<String>

Specifies the path within a Host Service provider to the hypervisor connection item to which to add the address. The path specified must be in one of the following formats: <drive>:\Connections\<HypervisorConnectionName> or <drive>:\Connections\{HHypervisorConnection Uid>.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>localhost</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.String

Get-HypXenServerAddress returns a list of strings containing the available address.

Notes

For this to work as required with HTTPS connections, the certificates installed on the XenServers must be trusted by all controllers. Typically, this means having the root certificate for the certificate trust chain installed on all controllers.Wildcard certificates are not supported for this operation.

In the case of failure, the following errors can be produced.

Error Codes

-------------

InputConnectionsPathInvalid

The path provided is not to an item in a subdirectory of a hosting unit item.

HypervisorConnectionNotXenServer

The hypervisor connection to which the path refers is not for a Citrix XenServer hypervisor.

HypervisorConnectionObjectNotFound
The hypervisor connection at the path specified cannot be located.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

DatabaseError

An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured

The operation could not be completed as the database for the service is not configured.

DataStoreException

An error occurred in the service whilst attempting a database operation - communication to database failed for various reasons.

CommunicationError

An error occurred whilst communicating with the service.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-HypXenServerAddress -Path XDHyp:\Connections\MyConnection

https:\myserver.com
https:\myServer1.com

gets the available addresses for the connection "MyConnection".

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-HypXenServerAddress -LiteralPath XDHyp:\Connections\MyConnection | Add-HypHypervisorConnectionAddress -Path XDHyp:\Connections\MyConnectionPath

adds all the available addresses for the XenServer pool in "MyConnection" to the hypervisor connection.
New-HypVMSnapshot

Oct 10, 2013
Creates a new snapshot for the specified VM item path.

Syntax


[-CommonParameters]

Detailed Description

This command provides a mechanism for creating a new snapshot of a virtual machine given a Host Service provider path to a VM. The resulting snapshot can then be used for operations that require a snapshot to work.

Related topics

Parameters

-LiteralPath <String>

Specifies the path within a hosting unit provider to the virtual machine item for which to create a new snapshot. The path specified must be in one of the following formats: <drive>:\Connections\<Connection Name>\<Item Path of VM object> or <drive>:\Connections\<Connection Uid>\<Item Path of VM object> or <drive>:\HostingUnits\<Hosting Unit Name>\<Item Path of VM object> or <drive>:\HostingUnits\<Hosting Unit Uid>\<Item Path of VM object>

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-SnapshotName <String>

The name to give the new snapshot. This will be visible in the hypervisor management console.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

-SnapshotDescription <String>

The description to add to the snapshot. This will be visible in the hypervisor management console.
Required? | false
---|---
Default Value |
Accept Pipeline Input? | false

**Input Type**

System.String You can pipe a string that contains a path to Get-HypConfigurationDataForItem

**Return Values**

System.String

The provider path to the newly created snapshot.

**Notes**

In the case of failure the following errors can be produced.

**Error Codes**

-------------

InputHypervisorItemPathInvalid

The path provided is not to an item in a subdirectory of a connection item or a hosting unit item.

InvalidHypervisorItemPath

No item exists with the specified path.

InvalidHypervisorItem

The item specified by the path exists, but is not a VM Item.

SnapshotNameAlreadyInUse

The specified name is already in use and would cause a name resolution clash.

FailedToCreateSnapshot

The snapshot creation process failed.

HypervisorInMaintenanceMode

The hypervisor is in maintenance mode.

DatabaseError

An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured

The operation could not be completed as the database for the service is not configured.

DataStoreException

An error occurred in the service whilst attempting a database operation - communication to database failed for various reasons.

CommunicationError

An error occurred whilst communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.2119
Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> New-HypVMSnapshot -LiteralPath XD hyp:\Connections\MyConnection\MyVm.vm -SnapshotName "New snapshot" -SnapshotDescription "Example snapshot"

XD hyp:\Connections\MyConnection\MyVm.vm\New snapshot.snapshot

This command creates a snapshot of a VM called 'MyVm.vm' within a hypervisor connection called 'MyConnection'.
Remove-HypHostingUnitMetadata

Oct 10, 2013
Removes metadata from the given HostingUnit.

Syntax

Remove-HypHostingUnitMetadata [-HostingUnitUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-HypHostingUnitMetadata [-HostingUnitUid] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-HypHostingUnitMetadata [-HostingUnitName] <String> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-HypHostingUnitMetadata [-HostingUnitName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-HypHostingUnitMetadata [-InputObject] <HostingUnit[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-HypHostingUnitMetadata [-InputObject] <HostingUnit[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given HostingUnit.

Related topics

Add-HypHostingUnitMetadata
Set-HypHostingUnitMetadata

Parameters

- **-HostingUnitUid** <Guid>
  
  Id of the HostingUnit
  
  Required? | true
  ---|---
  Default Value
  Accept Pipeline Input? | true (ByValue, ByPropertyName)

- **-HostingUnitName** <String>
  
  Name of the HostingUnit
  
  Required? | true
<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-InputObject</code>&lt;HostingUnit[]&gt;</td>
<td></td>
</tr>
<tr>
<td>Objects to which the metadata is to be added.</td>
<td></td>
</tr>
</tbody>
</table>

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

| `-Map`<PSObject> |
| Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed. |

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

| `-Name`<String> |
| The metadata property to remove. |

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

| `-LoggingId`<Guid> |
| Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets. |

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

| `-AdminAddress`<String> |
| |

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.2122
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

Error Codes

---------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.
ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-HypHostingUnit | % { Remove-HypHostingUnitMetadata -Map $_.MetadataMap }
Remove all metadata from all HostingUnit objects.
Remove-HypHostingUnitNetwork

Oct 10, 2013
Removes network from a hosting unit.

Syntax

Remove-HypHostingUnitNetwork [-LiteralPath] <String> [-NetworkPath] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Lets you remove networks from a hosting unit. This does not remove the network from the hypervisor just the reference to the network for the Host Service. After it is removed, the network is no longer available for associating with virtual NICs (when creating new virtual machines with the Machine Creation Services). Any virtual machines that have already been created continue to use the network until they are removed from the deployment. This command cannot be used if the connection for the hosting unit is in maintenance mode. If the network to be removed no longer exists on the hypervisor for the hosting unit, you must supply a fully qualified path to the network location.

Related topics

Add-HypHostingUnitNetwork

Parameters

-LiteralPath <String>

Required? true
Default Value
Accept Pipeline Input? true

-NetworkPath <String>

Required? true
Default Value
Accept Pipeline Input? true (ByValue)

-LoggingId <Guid>

Required? false
Default Value
Accept Pipeline Input? false

-AdminAddress <String>

Required? false
Default Value localhost. Once a value is provided by any cmdlet, this value becomes the default.
Accept Pipeline Input? false

Input Type

System.String You can pipe a string the contains a path to Remove-HypHostingUnitNetwork (Path parameter).
Notes

In the case of failure, the following errors can be produced.

Error Codes
-------------
HostingUnitsPathInvalid
The path provided is not to an item in a subdirectory of a hosting unit item.
HostingUnitNetworkObjectToDeleteDoesNotExist
The network path specified is not part of the hosting unit.
HypervisorInMaintenanceMode
The hypervisor for the connection is in maintenance mode.
DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
DataStoreException
An error occurred in the service while attempting a database operation. Communication to the database failed for various reasons.
CommunicationError
An error occurred while communicating with the service.
PermissionDenied
The user does not have administrative rights to perform this operation.
ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Remove-HypHostingUnitNetwork -LiteralPath XDHyp:\HostingUnits\MyHostingUnit -NetworkPath 'XDHyp:\HostingUnits\MyHostingUnits\newNetwork.network'
The command removes the network called "newNetwork.network" from the hosting unit called "MyHostingUnit"
Remove-HypHostingUnitStorage

Oct 10, 2013
Removes storage from a hosting unit.

Syntax


Detailed Description

Lets you remove storage locations from a hosting unit. This does not remove the storage from the hypervisor, just the reference to the storage for the Host Service. After it is removed, the storage is no longer used to store hard disks (when creating new virtual machines with the Machine Creation Services). The hard disks already located on the storage remain in place and virtual machines that have already been created continue to use the storage until they are removed from the deployment. This command cannot be used if the connection for the hosting unit is in maintenance mode. If the storage location to be removed no longer exists on the hypervisor for the hosting unit, you must supply a fully qualified path to the storage location.

Related topics

Add-HypHostingUnitStorage

Parameters

-LiteralPath:<String>

Required? true
Default Value
Accept Pipeline Input? false

-StoragePath:<String>

Required? false
Default Value
Accept Pipeline Input? true (ByValue)

-StorageType:<StorageType>

Required? false
Default Value OSStorage
Accept Pipeline Input? false

-LoggingId:<Guid>

Required? false
Default Value
Accept Pipeline Input? false

Parameters

-LiteralPath:<String>

Required? true
Default Value
Accept Pipeline Input? false

-StoragePath:<String>

Required? false
Default Value
Accept Pipeline Input? true (ByValue)

-StorageType:<StorageType>

Required? false
Default Value OSStorage
Accept Pipeline Input? false

-LoggingId:<Guid>

Required? false
Default Value
Accept Pipeline Input? false
-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. This can be a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

System.String You can pipe a string the contains a path to Remove-HypHostingUnitStorage (Path parameter).

Notes

After storage is removed, it is the administrator's responsibility to maintain its contents. The Citrix XenDesktop Machine Creation Services do not try to clean up any data that is stored in the storage location.

If all storage is removed from the hosting unit, other features of the Machine Creation Services stop functioning until some storage is added again.

In the case of failure, the following errors can be produced.

Error Codes

----------

HostingUnitsPathInvalid
The path provided is not to an item in a subdirectory of a hosting unit item.

HostingUnitStorageObjectToDeleteDoesNotExist
The storage path specified is not part of the hosting unit.

HypervisorInMaintenanceMode
The hypervisor for the connection is in maintenance mode.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation. Communication to the database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------- EXAMPLE 1 --------------------
c:\PS> Remove-HypHostingUnitStorage -LiteralPath XDhypes\\HostingUnits\MyHostingUnit -StoragePath 'XDhype\\HostingUnits\MyHostingUnits\newStorage.storage'
The command removes the OS storage location called "newStorage.storage" from the hosting unit called "MyHostingUnit"
-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-ChildItem XDHYP:\HostingUnits\Host\*.storage | Remove-HypHostingUnitStorage XDHYP:\HostingUnits\Host1
The command removes all OS storage from all of the hosting unit called "Host1".

-------------------------- EXAMPLE 3 --------------------------

c:\PS>Get-ChildItem XDHYP:\HostingUnits\Host\*.storage | Remove-HypHostingUnitStorage -StorageType PersonalvDiskStorage
The command removes all PersonalvDisk storage from all of the hosting unit called "Host1".
Remove-HypHypervisorConnectionAddress

Oct 10, 2013
Removes addresses from the list of available connection addresses.

Syntax
Remove-HypHypervisorConnectionAddress [-LiteralPath] <String> [-HypervisorAddress] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters>

Detailed Description
Provides the ability to remove addresses that can be used to connect to the hypervisor specified by the hypervisor connection. If all addresses are removed, the connection will not be able to be used until a valid connection is added to the hypervisor connection.

Related topics
Add-HypHypervisorConnectionAddress

Parameters
-LiteralPath <String>
Specifies the path within a Host Service provider to the hosting unit item to which to add the address. The path specified must be in one of the following formats: <drive>:\HostingUnits\<HostingUnitName> or <drive>:\HostingUnits\{HostingUnit Uid}.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-HypervisorAddress <String>
Specifies the address to be removed. If this parameter is not provided then all addresses will be removed from the hypervisor connection.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value will become the default.</td>
<td>false</td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | false

Input Type
System:string You can pipe a string that contains a path to Remove-HypHypervisorConnectionAddress (Path parameter).

Notes
Alterations to a hypervisor connection will affect all entities that reference the connection. If all addresses are removed from the connection, any other entities that reference the hypervisor connection (e.g. hosting units) will not be able to make new connections to the hypervisor.

In the case of failure the following errors can be produced.

Error Codes
-----------

InputConnectionsPathInvalid
The path provided is not to an item in a subdirectory of a hosting unit item.

HypervisorConnectionAddressForeignKeyObjectDoesNotExist
The hypervisor connection to which the address is to be added could not be located.

HypervisorInMaintenanceMode
The hypervisor for the connection is in maintenance mode.

DatabaseError
An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured
The operation could not be completed as the database for the service is not configured.

DataStoreException
An error occurred in the service whilst attempting a database operation - communication to database failed for various reasons.

CommunicationError
An error occurred whilst communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Remove-HypHypervisorConnectionAddress -LiteralPath XDHyp:\HostingUnits\MyHypervisorConnection -HypervisorAddress 'http:\myserver.com'
The command removes the address "http:\myserver.com" from the hypervisor connection called "MyHypervisorConnection".

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Get-Item -Path XDHYP:\Connections\* | Remove-HypHypervisorConnectionAddress
The command removes all addresses from all the hypervisor connections currently defined.
Remove-HypHypervisorConnectionMetadata

Oct 10, 2013
Removes metadata from the given HypervisorConnection.

Syntax

Remove-HypHypervisorConnectionMetadata [-HypervisorConnectionUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-HypHypervisorConnectionMetadata [-HypervisorConnectionUid] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-HypHypervisorConnectionMetadata [-HypervisorConnectionName] <String> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-HypHypervisorConnectionMetadata [-HypervisorConnectionName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-HypHypervisorConnectionMetadata [-InputObject] <HypervisorConnection[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-HypHypervisorConnectionMetadata [-InputObject] <HypervisorConnection[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given HypervisorConnection.

Related topics

Add-HypHypervisorConnectionMetadata
Set-HypHypervisorConnectionMetadata

Parameters

-HypervisorConnectionUid <Guid>
Id of the HypervisorConnection

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-HypervisorConnectionName <String>
Name of the HypervisorConnection

| Required? | true |
- **InputObject** `<HypervisorConnection[]>`  
  Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Map** `<PSObject>`  
  Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with `new-object "System.Collections.Generic.Dictionary[String,String]"`). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name** `<String>`  
  The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

- **LoggingId** `<Guid>`  
  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>

- **AdminAddress** `<String>`
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.
ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-HypHypervisorConnection | % { Remove-HypHypervisorConnectionMetadata -Map $_.MetadataMap }
Remove all metadata from all HypervisorConnection objects.
Remove-HypHypervisorConnectionScope

Oct 10, 2013
Remove the specified HypervisorConnection(s) from the given scope(s).

Syntax

Remove-HypHypervisorConnectionScope [-Scope] <String[]> -InputObject <HypervisorConnection[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-HypHypervisorConnectionScope [-Scope] <String[]> -HypervisorConnectionUid <Guid[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-HypHypervisorConnectionScope [-Scope] <String[]> -HypervisorConnectionName <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The RemoveHypHypervisorConnectionScope cmdlet is used to remove one or more HypervisorConnection objects from the given scope(s).

There are multiple parameter sets for this cmdlet, allowing you to identify the HypervisorConnection objects in different ways:

- HypervisorConnection objects can be piped in or specified by the InputObject parameter
- The HypervisorConnectionUid parameter specifies objects by HypervisorConnectionUid
- The HypervisorConnectionName parameter specifies objects by HypervisorConnectionName (supports wildcards)

To remove a HypervisorConnection from a scope you need permission to change the scopes of the HypervisorConnection.

If the HypervisorConnection is not in a specified scope, that scope will be silently ignored.

Related topics

Add-HypHypervisorConnectionScope

Get-HypScopedObject

Parameters

- **-Scope** <String[]>

  Specifies the scopes to remove the objects from.

<table>
<thead>
<tr>
<th>Required</th>
<th>Default Value</th>
<th>Accept Pipeline Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-InputObject** <HypervisorConnection[]>

  Specifies the HypervisorConnection objects to be removed.

<table>
<thead>
<tr>
<th>Required</th>
<th>Default Value</th>
<th>Accept Pipeline Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>
### -HypervisorConnectionUid

Specifies the HypervisorConnection objects to be removed by HypervisorConnectionUid.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

### -HypervisorConnectionName

Specifies the HypervisorConnection objects to be removed by HypervisorConnectionName.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

### -LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### -AdminAddress

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

None

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

UnknownObject

One of the specified objects was not found.
ScopeNotFound
One of the specified scopes was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

----------------------------- EXAMPLE 1 -----------------------------
c:\PS>Remove-HypHypervisorConnectionScope Finance -HypervisorConnectionUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
Removes a single HypervisorConnection from the 'Finance' scope.

----------------------------- EXAMPLE 2 -----------------------------
c:\PS>Remove-HypHypervisorConnectionScope Finance,Marketing -HypervisorConnectionUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
Removes a single HypervisorConnection from multiple scopes.

----------------------------- EXAMPLE 3 -----------------------------
c:\PS>Get-HypHypervisorConnection | Remove-HypHypervisorConnectionScope Finance
Removes all visible HypervisorConnection objects from the 'Finance' scope.

----------------------------- EXAMPLE 4 -----------------------------
c:\PS>Remove-HypHypervisorConnectionScope Finance -HypervisorConnectionName A*
Removes HypervisorConnection objects with a name starting with an 'A' from the 'Finance' scope.
Remove-HypMetadata

Oct 10, 2013
Removes metadata from a hypervisor connection or hosting unit.

Syntax
[<CommonParameters>]

Detailed Description
Provides the ability for custom data to be removed from a specific hosting unit or hypervisor connection. If the Property parameter is not provided, all metadata will be removed from the specified item.

Related topics
Remove-HypMetadata

Parameters
-LiteralPath<String>
Specifies the path within a Host Service provider to the hosting unit or hypervisor connection item from which to remove the metadata. The path specified must be in one of the following formats: <drive>\HostingUnits\<HostingUnitName> or <drive>\HostingUnits\{<HostingUnit Uid> or <drive>\Connections\<Connection Name> or <drive>\Connections\{<Connection Uid>}

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Property<String>
Specifies the property name of the metadata to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId<Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.
<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**: `<String>`

Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

**Input Type**

System.String You can pipe a string that contains a path to Add-HypMetadata (Path parameter).

**Notes**

In the case of failure the following errors can be produced:

**Error Codes**

`---------`

InvalidPath

The path provided is not in the required format.

HostingUnitMetadataObjectToDeleteDoesNotExist

The hosting unit supplied in the path does not exist.

HypervisorConnectionObjectToDeleteDoesNotExist

The hypervisor connection supplied in the path does not exist.

MetadataContainerUndefined

The specified path does not reference a hosting unit or a hypervisor connection.

DatabaseError

An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured

The operation could not be completed as the database for the service is not configured.

DataStoreException


An error occurred in the service whilst attempting a database operation - communication to database failed for various reasons.

CommunicationError

An error occurred whilst communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Remove-HypMetadata -LiteralPath XDHyp:\Connections\MyConnection -Property MyProperty
The command removes the metadata with the property "MyProperty" from the hypervisor connection called "MyConnection".

-------------------------- EXAMPLE 2 --------------------------

C:\PS>Remove-HypMetadata -LiteralPath XDHyp:\Connections\MyConnection
The command removes all the metadata from the hypervisor connection called "MyConnection".

-------------------------- EXAMPLE 3 --------------------------

C:\PS>dir XDHyp:\connections | Remove-HypMetadata
The command removes all the metadata from all the hypervisor connections.
Remove-HypServiceMetadata

Oct 10, 2013
Removes metadata from the given Service.

Syntax

Remove-HypServiceMetadata [-ServiceHostId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-HypServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-HypServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-HypServiceMetadata [-InputObject] <Service[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Provides the ability to remove metadata from the given Service.

Related topics

Set-HypServiceMetadata

Parameters

-ServiceHostId <Guid>

Id of the Service

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-InputObject <Service[]>

Objects to which metadata is to be removed.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue) |

-Name <String>
The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Map**<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

Error Codes

----------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> Get-HypService | % { Remove-HypServiceMetadata -Map $_.MetadataMap } 
Remove all metadata from all Service objects.
Reset-HypServiceGroupMembership

Oct 10, 2013
Reloads the access permissions and configuration service locations for the Host Service.

Syntax

Reset-HypServiceGroupMembership [-ConfigServiceInstance] <ServiceInstance[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Enables you to reload Host Service access permissions and configuration service locations. The Reset-HypServiceGroupMembership command must be run on at least one instance of the service type (Hyp) after installation and registration with the configuration service. Without this operation, the Host services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-HypServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Related topics

Parameters

-ConfigServiceInstance <ServiceInstance[]>
  Specifies the configuration service instance object that represents the service instance for the type 'InterService' that references a configuration service for the deployment.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type
Citrix.Host.Sdk.ServiceInstance[] Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-HypServiceGroupMembership command.

Notes
If the command fails, the following errors can be returned.

Error Codes
---------

NoSuitableServiceInstance
None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-HypServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config -AdminAddress OtherServer.example.com | Reset-HypServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.2146
Set-HypAdminConnection

Oct 10, 2013
Set the controller that will be used by the cmdlets that form the Host Service PowerShell snap-in.

Syntax

Set-HypAdminConnection [-AdminAddress <String>] [CommonParameters]

Detailed Description

Provides the ability to set the default controller address that will be used by the cmdlets in order to communicate with the controller. Most Host Service cmdlets take an 'AdminAddress' parameter that can be used to define the controller (when used this will override this setting). However the Set-Location cmdlet in the Host Service provider cannot offer this option and the controller address must be set using this cmdlet if no other cmdlets have been used to set the address previously during the use of the current runspace.

Related topics

Parameters

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Set-HypAdminConnection -AdminAddress myserver.com

This command sets the controller address for the commands to be "myserver.com". All commands run will use this address until it is altered either by another call to this command, or the use of the 'AdminAddress' parameter on another command in the Host Service snap-in.
Set-HypDBConnection

Oct 10, 2013
Configures a database connection for the Host Service.

Syntax

Set-HypDBConnection [-DBConnection] <String> [-Force] [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Configures a connection to a database in which the Host Service can store its state. The service will attempt to connect and start using the database immediately after the connection is configured. The database connection string is updated to the specified value regardless of whether it is valid or not. Specifying an invalid connection string prevents a service from functioning until the error is corrected.

After a connection is configured, you cannot alter it without first clearing it (by setting the connection to $null).

You do not need to configure a database connection to use this command.

Related topics

Get-HypServiceStatus
Get-HypDBConnection
Test-HypDBConnection

Parameters

-**DBConnection**<String>

Specifies the database connection string to be used by the Host Service. Passing in $null will clear any existing database connection configured.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

-**Force**<SwitchParameter>

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**: String

  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**


The Set-HypDBConnection command returns an object containing the status of the Host Service together with extra diagnostics information.

**DBUnconfigured**

The Host Service does not have a database connection configured.

**DBRejectedConnection**

The database rejected the logon attempt from the Host Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**

The expected stored procedures are missing from the database. This may be because the Host Service schema has not been added to the database.

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**

The version of the Host Service currently in use is incompatible with the version of the Host Service schema on the database. Upgrade the Host Service to a more recent version.

**DBOlderVersionThanService**

The version of the Host Service schema on the database is incompatible with the version of the Host Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**

A database schema upgrade is currently in progress.

**OK**

The Host Service is running and is connected to a database containing a valid schema.

**Failed**

The Host Service has failed.

**Unknown**
The status of the Host Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured. After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Set-HypDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK

Configures a database connection string for the Host Service.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Set-HypDBConnection -DBConnection "Invalid Connection String"

Invalid database connection string format.

Configures an invalid database connection string for the Host Service.
Set-HypHostingUnitMetadata

Oct 10, 2013
Adds or updates metadata on the given HostingUnit.

Syntax

Set-HypHostingUnitMetadata [-HostingUnitUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-HypHostingUnitMetadata [-HostingUnitUid] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-HypHostingUnitMetadata [-HostingUnitName] <String> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-HypHostingUnitMetadata [-HostingUnitName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-HypHostingUnitMetadata [-InputObject] <HostingUnit[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-HypHostingUnitMetadata [-InputObject] <HostingUnit[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability for additional custom data to be stored against given HostingUnit objects.

Related topics
Add-HypHostingUnitMetadata
Remove-HypHostingUnitMetadata

Parameters

-HostingUnitUid <Guid>
Id of the HostingUnit

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-HostingUnitName <String>
Name of the HostingUnit

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>-InputObject</code> <code>&lt;HostingUnit[]&gt;</code></td>
<td>Objects to which the metadata is to be added.</td>
</tr>
<tr>
<td><code>-Map</code> <code>&lt;PSObject&gt;</code></td>
<td>Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with <code>@{&quot;name1&quot; = &quot;val1&quot;; &quot;name2&quot; = &quot;val2&quot;}</code>) or a string dictionary (created with <code>new-object &quot;System.Collections.Generic.Dictionary[String,String]&quot;</code>).</td>
</tr>
<tr>
<td><code>-Name</code> <code>&lt;String&gt;</code></td>
<td>Specifies the property name of the metadata to be added. The property must be unique for the HostingUnit specified. The property cannot contain any of the following characters `/:*?&lt;&gt;</td>
</tr>
<tr>
<td><code>-Value</code> <code>&lt;String&gt;</code></td>
<td>Specifies the value for the property.</td>
</tr>
<tr>
<td><code>-LoggingId</code> <code>&lt;Guid&gt;</code></td>
<td>Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.</td>
</tr>
</tbody>
</table>

https://docs.citrix.com   © 1999-2017 Citrix Systems, Inc. All rights reserved.   p.2152
Default Value

Accept Pipeline Input? false

AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

Required? false

Default Value
Localhost. Once a value is provided by any cmdlet, this value becomes the default.

Accept Pipeline Input? false

Return Values


Set-HypHostingUnitMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>

Specifies the name of the property.

Value <string>

Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.
PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Set-HypHostingUnitMetadata -HostingUnitUid 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the HostingUnit with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Set-HypHostingUnitStorage

Oct 10, 2013
Sets options for a storage location on a hosting unit.

Syntax

Detailed Description
This command lets you set options for storage locations that are defined for a hosting unit. This command cannot be used if the connection for the hosting unit is in maintenance mode.

Related topics
Parameters
-LiteralPath:String>
Specifies the path to the hosting unit which defines the storage. The path must be in one of the following formats: <drive>:\HostingUnits\<HostingUnitName> or <drive>:\HostingUnits\{<HostingUnit Uid>}

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-StoragePath:<String>
Specifies the path to the storage that will be modified. The path must be in one of the following formats: <drive>:\Connections\<ConnectionName>\MyStorage.storage or <drive>:\Connections\{<Connection Uid>\}\MyStorage.storage or <drive>:\HostingUnits\<HostingUnitName>\MyStorage.storage or <drive>:\HostingUnits\{<hostingUnit Uid>\}\MyStorage.storage

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-StorageType:StorageType>
Specifies the type of storage in StoragePath. Supported storage types are: OSStorage PersonalvDiskStorage

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>OSStorage</td>
</tr>
</tbody>
</table>

-Superseded:<Boolean>
Specifies that this storage has been superseded and should not be used for future provisioning operations. Existing virtual machines using this storage will continue to function.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

-LoggingId:<Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects. This can be a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. When a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input?  false

Input Type

System.String

You can pipe a string that contains a path to Add-HypHostingUnitStorage (StoragePath parameter).

Return Values

Citrix.XDPowerShell.HostingUnit

Add-HypHostingUnitStorage returns an object containing the new definition of the hosting unit.

HostingUnitId <Guid>

Specifies the unique identifier for the hosting unit.

HostingUnitName <string>

Specifies the name of the hosting unit.

HypervisorConnection <Citrix.XDPowerShell.HypervisorConnection>

Specifies the connection that the hosting unit uses to access a hypervisor.

RootId <string>

Identifies, to the hypervisor, the root of the hosting unit.

RootPath <string>

The hosting unit provider path that represents the root of the hosting unit.

Storage <Citrix.XDPowerShell.Storage[]>

The list of storage items that the hosting unit can use.

PersonalDiskStorage <Citrix.XDPowerShell.Storage[]>

The list of storage items that the hosting unit can use for storing personal data.

VMTaggingEnabled <Boolean>

Specifies whether the metadata in the hypervisor can be used to store information about the XenDesktop Machine Creation Services.

NetworkId <string>

The hypervisor's internal identifier that represents the network specified for the hosting unit.

NetworkPath <string>

The hosting unit provider path to the network specified for the hosting unit.

Metadata <Citrix.XDPowerShell.Metadata[]>

A list of key value pairs that can store additional information about the hosting unit.

Notes

The storage path must be valid for the hosting unit. The rules that are applied are as follows: XenServer (HypervisorConnection Type = XenServer)

NA

VMWare vSphere/ESX (HypervisorConnection Type = vCenter)

The storage path must be directly contained in the root path item of the hosting unit.

Microsoft SCVMM/Hyper-v (HypervisorConnection Type = SCVMM)
Only one storage entry for these connection types is valid, and it must reference an SMB share. Additionally, if a Hyper-V failover cluster is used the SMB share must be the top-level mount point of the cluster shared volume on one of the servers in the cluster (i.e. C:\ClusterStorage).

In the case of failure, the following errors can be produced:

Error Codes
-----------
HostingUnitPathInvalid
The path provided is not to an item in a subdirectory of a hosting unit item.
HostingUnitStoragePathInvalid
The specified path is invalid.
HostingUnitStoragePathInvalid
The storage path cannot be found or is invalid. See notes above about validity.
HostingUnitStorageDuplicateObjectExists
The specified storage path is already part of the hosting unit.
HypervisorInMaintenanceMode
The hypervisor for the connection is in maintenance mode.
DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
DataStoreException
An error occurred in the service while attempting a database operation. Communication to the database failed for various reasons.
CommunicationError
An error occurred while communicating with the service.
PermissionDenied
The user does not have administrative rights to perform this operation.
ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS> Set-HypHostingUnitStorage -LiteralPath XDHyp:\HostingUnits\MyHostingUnit -StoragePath 'XDHyp:\HostingUnits\MyHostingUnits\newStorage.storage' -Superseded $true

HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
HostingUnitName      : MyHostingUnit
HypervisorConnection : MyConnection
RootPath             : /
RootId               :
NetworkPath          : /Network 0.network
NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
Storage              : {Local storage.storage, newStorage.storage}
PersonalvDiskStorage : {newStorage.storage}
VMTaggingEnabled     : True
Metadata             : {}

The command updates a storage location called "newStorage.storage" associated with the hosting unit called "MyHostingUnit."
Set-HypHypervisorConnectionMetadata

Oct 10, 2013
Adds or updates metadata on the given HypervisorConnection.

Syntax
Set-HypHypervisorConnectionMetadata [-HypervisorConnectionUid] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-HypHypervisorConnectionMetadata [-HypervisorConnectionUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-HypHypervisorConnectionMetadata [-HypervisorConnectionName] <String> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-HypHypervisorConnectionMetadata [-HypervisorConnectionName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-HypHypervisorConnectionMetadata [-InputObject] <HypervisorConnection[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-HypHypervisorConnectionMetadata [-InputObject] <HypervisorConnection[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Provides the ability for additional custom data to be stored against given HypervisorConnection objects.

Related topics
Add-HypHypervisorConnectionMetadata
Remove-HypHypervisorConnectionMetadata

Parameters
-HypervisorConnectionUid <Guid>
Id of the HypervisorConnection

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-HypervisorConnectionName <String>
Name of the HypervisorConnection

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-InputObject <HypervisorConnection[]>
Objects to which the metadata is to be added.

| Required? | true |
- **Name**: <String>
  
  Specifies the property name of the metadata to be added. The property must be unique for the HypervisorConnection specified. The property cannot contain any of the following characters: `\/:.*=?<>|[]`.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Value**: <String>
  
  Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Map**: <PSObject>
  
  Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**: <Guid>
  
  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**: <String>
  
  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**

Set-HypHypervisorConnectionMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>

Specifies the name of the property.

Value <string>

Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> Set-HypHypervisorConnectionMetadata -HypervisorConnectionUid 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Add metadata with a name of 'property' and a value of 'value' to the HypervisorConnection with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Set-HypServiceMetadata

Oct 10, 2013
Adds or updates metadata on the given Service.

Syntax

Set-HypServiceMetadata [-ServiceHostId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-HypServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-HypServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-HypServiceMetadata [-InputObject] <Service[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Allows you to store additional custom data against given Service objects.

Related topics

Remove-HypServiceMetadata

Parameters

-servicehostid <Guid>

Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-inputobject <Service[]>

Objects to which metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-name <String>
Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters `/:.*?<>|\[]()"'

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Value <String>**

Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Map <PSObject>**

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with `new-object "System.Collections.Generic.Dictionary[String,String]"`).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-LoggingId <Guid>**

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress <String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


Set-HypServiceMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>

Specifies the name of the property.

Value <string>

Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

--------

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> Set-HypServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2A06DD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DD0805C'.
Start-HypVM

Oct 10, 2013
Starts a VM.

Syntax

Start-HypVM [-LiteralPath <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
This command provides a mechanism to start a VM.

Related topics

Parameters

-LiteralPath <String>
Specifies the path within a hosting unit provider to the virtual machine item to start. The path specified must be in one of the following formats: <drive>:\Connections\<Connection Name>\<Item Path of VM object> or <drive>:\Connections\{<connection Uid>\<Item Path of VM object>} or <drive>:\HostingUnits\<HostingUnit Name>\<Item Path of VM object> or <drive>:\HostingUnits\{<hostingUnit Uid>\<Item Path of VM object>}

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

System\string You can pipe a string the contains a path.

Return Values

System\void.

Notes
In the case of failure the following errors can be produced.
Error Codes

---------

InputHypervisorItemPathInvalid
The path provided is not to an item in a sub directory of a connection item or a hosting unit item.

InvalidHypervisorItemPath
No item exists with the specified path.

InvalidHypervisorItem
The item specified by the path exists, but is not a VM Item.

HypervisorInMaintenanceMode
The hypervisor is in maintenance mode.

DatabaseError
An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured
The operation could not be completed as the database for the service is not configured.

CommunicationError
An error occurred whilst communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Start-HypVM -LiteralPath XDHyp:\Connections\MyConnection\MyVm.vm
This command starts a VM called 'MyVm.vm' within a hypervisor connection called 'MyConnection'.
Stop-HypVM

Oct 10, 2013
Stops a VM by issuing a Shutdown request

Syntax

Stop-HypVM [-LiteralPath] <String> [-AdminAddress <String>] [CommonParameters]

Detailed Description

This command provides a mechanism to change the power state of a VM from running to stopped.

Related topics

Parameters

-LiteralPath <String>

Specifies the path within a hosting unit provider to the virtual machine item to stop. The path specified must be in one of the following formats: <drive>:\Connections\<Connection Name>\<Item Path of VM object> or <drive>:\Connections\{<connection Uid>\<Item Path of VM object>} or <drive>:\HostingUnits\<HostingUnit Name>\<Item Path of VM object> or <drive>:\HostingUnits\{<hostingUnit Uid>\<Item Path of VM object>}

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value will become the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

System: string You can pipe a string containing a path.

Return Values

System: void.

Notes

In the case of failure, the following errors can be produced.
Error Codes
----------

InputHypervisorItemPathInvalid
The path provided is not to an item in a sub directory of a connection item or a hosting unit item.

InvalidHypervisorItemPath
No item exists with the specified path.

InvalidHypervisorItem
The item specified by the path exists, but is not a VM Item.

HypervisorInMaintenanceMode
The hypervisor is in maintenance mode.

DatabaseError
An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured
The operation could not be completed as the database for the service is not configured.

CommunicationError
An error occurred whilst communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------

C:\PS>Stop-HypVM -LiteralPath XDHyp:\Connections\MyConnection\MyVm.vm
This command stops a VM called 'MyVm.vm' within a hypervisor connection called 'MyConnection'.
Test-HypDBConnection

Oct 10, 2013
Tests a database connection for the Host Service.

Syntax
Test-HypDBConnection [-DBConnection] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description
Tests a connection to the database in which the Host Service can store its state. The service will attempt to connect to the database without affecting the current connection to the database.

You do not have to clear the connection to use this command.

Related topics
Get-HypServiceStatus
Get-HypDBConnection
Set-HypDBConnection

Parameters
-DBConnection <String>
Specifies the database connection string to be tested by the Host Service.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
The Test-HypDBConnection command returns an object containing the status of the Host Service if the connection string of the specified data store were to be set to the string being tested, together with extra diagnostics information for the specified connection string.

**DBRejectedConnection**

The database rejected the logon attempt from the Host Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**

The expected stored procedures are missing from the database. This may be because the Host Service schema has not been added to the database.

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**

The version of the Host Service currently in use is incompatible with the version of the Host Service schema on the database. Upgrade the Host Service to a more recent version.

**DBOlderVersionThanService**

The version of the Host Service schema on the database is incompatible with the version of the Host Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**

A database schema upgrade is currently in progress.

**OK**

The Set-HypDBConnection command would succeed if it were executed with the supplied connection string.

**Failed**

The Host Service has failed.

**Unknown**

The status of the Host Service cannot be determined.

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

-------------

**InvalidDBConnectionString**

The database connection string has an invalid format.

**PermissionDenied**

You do not have permission to execute this command.

**AuthorizationError**

There was a problem communicating with the Citrix Delegated Administration Service.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Test-HypDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Tests a database connection string for the Host Service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Test-HypDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Tests an invalid database connection string for the Host Service.
Test-HypHostingUnitNameAvailable

Oct 10, 2013
Checks to see if the proposed name for a hosting unit is unused.

Syntax

Test-HypHostingUnitNameAvailable -HostingUnitName <String[]> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Checks to see if the proposed name for a hosting unit is unused. This check is done without regard for scoping of existing hosting unit, so even the names of inaccessible hosting units will be checked.

Related topics

New-Item
Rename-Item

Parameters

-HostingUnitName <String[]>
The name or names of the hosting units(s) to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

object[]
An array of PSObjects which pair the name and availability of the name

Notes

In the case of failure the following errors can be produced.
Error Codes

-----------

DatabaseError
An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured
The operation could not be completed as the database for the service is not configured.

DataStoreException
An error occurred in the service whilst attempting a database operation - communication to database failed for various reasons.

CommunicationError
An error occurred whilst communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------- EXAMPLE 1 --------------------------

Test-HypHostingUnitNameAvailable -HostingUnitName $NewHostingUnitName
This will test whether the if the value of $NewHostingUnitName could be used to create a new hosting unit or rename an existing one without failing because the name is not unique, returning true if the name is good.
Test-HypHypervisorConnectionNameAvailable

Oct 10, 2013
Checks to see if the proposed name for a hypervisor connection is unused.

Syntax

Test-HypHypervisorConnectionNameAvailable -HypervisorConnectionName <String[]> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Checks to see if the proposed name for a hypervisor connection is unused. This check is done without regard for scoping of existing hypervisor connection, so even the names of inaccessible hypervisor connection will be checked.

Related topics

New-Item
Rename-Item

Parameters

-HypervisorConnectionName <String[]>

The name or names of the hypervisor connection(s) to be tested.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values

object[]

An array of PSObjects which pair the name and availability of the name

Notes
In the case of failure the following errors can be produced.

Error Codes
-----------

DatabaseError
An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured
The operation could not be completed as the database for the service is not configured.

DataStoreException
An error occurred in the service whilst attempting a database operation - communication to database failed for various reasons.

CommunicationError
An error occurred whilst communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------

Test-HypHypervisorConnectionNameAvailable -HypervisorConnectionName $NewConnectionName
This will test whether the value of $NewConnectionName could be used to create a new hypervisor connection or rename an existing one without failing because the name is not unique, returning true if the name is good.
Update-HypHypervisorConnection

Oct 10, 2013
Requests the host service to update the connection properties that depend on the version of hypervisor in use.

Syntax
Update-HypHypervisorConnection [-LiteralPath] <String> [-LoggingId <Guid>] [-AdminAddress <String>] 
[<CommonParameters>]

Detailed Description
Allows the version specific properties of a hypervisor connection to be updated after an upgrade to the hypervisor system which may provide new capabilities.

Related topics
Parameters

-LiteralPath<String>
Specifies the path within a Host Service provider to the hypervisor connection item to be updated. The path specified must be in one of the following formats: <drive>:\Connections\<HypervisorConnectionName> or <drive>:\Connections\{HypervisorConnection Uid>

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-LoggingId<Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>
Specifies the address of a XenDesktop controller that the PowerShell snap-in will connect to. This can be provided as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
Default Value | LocalHost. Once a value is provided by any cmdlet, this value will become the default.
--- | ---
Accept Pipeline Input? | false

Notes
In the case of failure the following errors can be produced.

Error Codes
--------

InvalidPath
The path provided is not in the required format.

HypervisorInMaintenanceMode
The hypervisor is in maintenance mode and cannot be updated.

DatabaseError
An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured
The operation could not be completed as the database for the service is not configured.

DataStoreException
An error occurred in the service whilst attempting a database operation - communication to database failed for various reasons.

CommunicationError
An error occurred whilst communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples
--------------- EXAMPLE 1 --------------------------
c:\PS>Update-HypHypervisorConnection -LiteralPath xdhyp:\connections\Connection1
This command requests that the properties of Connection1 be updated to match the current capabilities of the hypervisor.
## Overview

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProvMachineCreationSnapin</td>
<td>The Machine Creation Service PowerShell snap-in provides administrative</td>
</tr>
<tr>
<td>Prov Filtering</td>
<td>Describes the common filtering options for XenDesktop cmdlets.</td>
</tr>
<tr>
<td>providers</td>
<td>Describes how Windows PowerShell providers provide access to data and</td>
</tr>
</tbody>
</table>

### Cmdlets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-ProvSchemeControllerAddress</td>
<td>Adds a list of host names (as DNS addresses) to a provisioning scheme.</td>
</tr>
<tr>
<td>Add-ProvSchemeMetadata</td>
<td>Adds metadata on the given ProvisioningScheme.</td>
</tr>
<tr>
<td>Add-ProvSchemeScope</td>
<td>Add the specified ProvisioningScheme(s) to the given scope(s).</td>
</tr>
<tr>
<td>Add-ProvTaskMetadata</td>
<td>Adds metadata on the given Task.</td>
</tr>
<tr>
<td>Get-ProvDBConnection</td>
<td>Gets the database string for the specified data store used by the MachineCreation Service.</td>
</tr>
<tr>
<td>Get-ProvDBSchema</td>
<td>Gets a script that creates the MachineCreation Service database schema for the specified data store.</td>
</tr>
<tr>
<td>Get-ProvDBVersionChangeScript</td>
<td>Gets a script that updates the MachineCreation Service database schema.</td>
</tr>
<tr>
<td>Get-ProvInstalledDBVersion</td>
<td>Gets a list of all available database schema versions for the MachineCreation Service.</td>
</tr>
<tr>
<td>Get-ProvObjectReference</td>
<td>Returns the number of local objects holding references to objects from other services.</td>
</tr>
<tr>
<td>Get-ProvScheme</td>
<td>Gets the list of provisioning schemes.</td>
</tr>
<tr>
<td>Get-ProvSchemeMasterVMImageHistory</td>
<td>Gets the list of master VM snapshots that have been used to provide hard disks to provisioning schemes.</td>
</tr>
<tr>
<td>Get-ProvScopedObject</td>
<td>Gets the details of the scoped objects for the MachineCreation Service.</td>
</tr>
<tr>
<td>Get-ProvService</td>
<td>Gets the service record entries for the MachineCreation Service.</td>
</tr>
<tr>
<td>Get-ProvServiceAddedCapability</td>
<td>Gets any added capabilities for the MachineCreation Service on the controller.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Get-ProvServiceInstance</td>
<td>Gets the service instance entries for the MachineCreation Service.</td>
</tr>
<tr>
<td>Get-ProvServiceStatus</td>
<td>Gets the current status of the MachineCreation Service on the controller.</td>
</tr>
<tr>
<td>Get-ProvTask</td>
<td>Gets the task history for the MachineCreation Service.</td>
</tr>
<tr>
<td>Get-ProvVM</td>
<td>Gets the VMs that were created using Citrix XenDesktop Machine Creation Services.</td>
</tr>
<tr>
<td>Lock-ProvVM</td>
<td>Locks a VM.</td>
</tr>
<tr>
<td>New-ProvScheme</td>
<td>Creates a new provisioning scheme.</td>
</tr>
<tr>
<td>New-ProvVM</td>
<td>Creates a new virtual machine.</td>
</tr>
<tr>
<td>Publish-ProvMasterVmImage</td>
<td>Update the master image associated with the provisioning scheme.</td>
</tr>
<tr>
<td>Remove-ProvScheme</td>
<td>Removes a provisioning scheme.</td>
</tr>
<tr>
<td>Remove-ProvSchemeControllerAddress</td>
<td>Removes metadata from a provisioning scheme.</td>
</tr>
<tr>
<td>Remove-ProvSchemeMasterVmImageHistory</td>
<td>Removes the history of provisioning scheme master image VMs.</td>
</tr>
<tr>
<td>Remove-ProvSchemeMetadata</td>
<td>Removes metadata from the given ProvisioningScheme.</td>
</tr>
<tr>
<td>Remove-ProvSchemeScope</td>
<td>Remove the specified ProvisioningScheme(s) from the given scope(s).</td>
</tr>
<tr>
<td>Remove-ProvServiceConfigurationData</td>
<td>Removes configuration data from the service.</td>
</tr>
<tr>
<td>Remove-ProvServiceMetadata</td>
<td>Removes metadata from the given Service.</td>
</tr>
<tr>
<td>Remove-ProvTask</td>
<td>Removes from the database completed tasks for the MachineCreation Service.</td>
</tr>
<tr>
<td>Remove-ProvTaskMetadata</td>
<td>Removes metadata from the given Task.</td>
</tr>
<tr>
<td>Remove-ProvVM</td>
<td>Removes virtual machines.</td>
</tr>
<tr>
<td>Rename-ProvScheme</td>
<td>Renames a provisioning scheme.</td>
</tr>
<tr>
<td>Reset-ProvServiceGroupMembership</td>
<td>Reloads the access permissions and configuration service locations for the MachineCreation Service.</td>
</tr>
<tr>
<td>Set-ProvDBConnection</td>
<td>Configures a database connection for the MachineCreation Service.</td>
</tr>
<tr>
<td>Set-ProvScheme</td>
<td>Changes the parameter values for a provisioning scheme.</td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set-ProvSchemeMetadata</td>
<td>Adds or updates metadata on the given ProvisioningScheme.</td>
</tr>
<tr>
<td>Set-ProvServiceConfigurationData</td>
<td>Sets the value for the given key in the service configuration data.</td>
</tr>
<tr>
<td>Set-ProvServiceMetadata</td>
<td>Adds or updates metadata on the given Service.</td>
</tr>
<tr>
<td>Set-ProvTaskMetadata</td>
<td>Adds or updates metadata on the given Task.</td>
</tr>
<tr>
<td>Stop-ProvTask</td>
<td>Stops currently running MachineCreation Service tasks.</td>
</tr>
<tr>
<td>Switch-ProvTask</td>
<td>Moves all MachineCreation Service tasks from the current execution host to another.</td>
</tr>
<tr>
<td>Test-ProvDBConnection</td>
<td>Tests a database connection for the MachineCreation Service.</td>
</tr>
<tr>
<td>Test-ProvSchemeNameAvailable</td>
<td>Checks to ensure that the proposed name for a provisioning scheme is unused.</td>
</tr>
<tr>
<td>Unlock-ProvScheme</td>
<td>Unlocks a Provisioning Scheme.</td>
</tr>
<tr>
<td>Unlock-ProvVM</td>
<td>Unlocks a VM.</td>
</tr>
</tbody>
</table>
about_ProvMachineCreationSnapin

Oct 10, 2013

TOPIC
about_ProvMachineCreationSnapin

SHORT DESCRIPTION
The Machine Creation Service PowerShell snap-in provides administrative functions for the Machine Creation Service.

COMMAND PREFIX
All commands in this snap-in have 'Prov' in their name.

LONG DESCRIPTION
The Machine Creation Service PowerShell snap-in enables both local and remote administration of the Machine Creation Service. It provides facilities to create virtual machines and manage the associated disk images.

The snap-in provides two main entities:

Provisioning Scheme

- Specifies details of new virtual machines created by the Machine Creation Service. Provisioning schemes define the following information.

  Hosting Unit
  Provides details of the hypervisor and storage on which new virtual machines will be created. Stored and maintained by the Host Service and PowerShell snap-in.

  Identity Pool
  Lists the Active Directory computer accounts available for use by new virtual machines. Stored and maintained by the Active Directory Identity Service and PowerShell snap-in.

Master Image
  Specifies the disk image that will be used for new virtual machines. Accessed through the hosting provider in the Host Service snap-in.

Provisioned VM

- Defines the virtual machines created by the Machine Creation Services. These virtual machines are associated with the provisioning scheme from
which they were created.

The processes of creating provisioning schemes and new virtual machines can take a significant amount of time to complete. For this reason, these long-running tasks can be run asynchronously so that other commands are accessible while the processes are running. Note, however, that only one long-running task can operate on a provisioning scheme at any one time. The processes are monitored using the Get-ProvTask command. For more information, see the help for Get-ProvTask.
about_Prov_Filtering

Oct 10, 2013

TOPIC

XenDesktop - Advanced Dataset Filtering

SHORT DESCRIPTION

Describes the common filtering options for XenDesktop cmdlets.

LONG DESCRIPTION

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

```
Get-<Noun> | Where { $_.Size = 'Small' } | Sort 'Date' | Select -First 10
```

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

```
Get-<Noun> -Size 'Small' -SortBy 'Date' -MaxRecordCount 10
```

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

- **-MaxRecordCount <int>**

  Specifies the maximum number of results to return.

  For example, to return only the first nine results use:

  ```
  Get-<Noun> -MaxRecordCount 9
  ```

  If not specified, only the first 250 records are returned, and if more are available, a warning is produced:
WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

To retrieve all records, specify a large number for -MaxRecordCount. As the value is an integer, you can use the following:

Get-<Noun> -MaxRecordCount [int]:::MaxValue

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount 
....

Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo : OperationStopped: (.) [Get-<Noun>], PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.
Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-sortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values. Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

-sortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'

-filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter 'Name -like "High**" -or (Priority -eq 1 -and Severity -ge 2)'

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

EXAMPLES

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company "citrix" -Product [X]EN"
Get-<Noun> -Product "Xen***" -Company "CITRIX"
Get-<Noun> -Filter { Company -eq 'Citrix' -and Product -like 'Xen***' }
See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```powershell
Get-<Noun> -Company "Abc["" # Matches Abc*
Get-<Noun> -Company "Abc"** # Matches Abc*
Get-<Noun> -Filter { Company -eq "Abc"** } # Matches Abc*
Get-<Noun> -Filter { Company -eq "A""B""C" } # Matches A"B"C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```powershell
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

```powershell
Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```powershell
Get-<Noun> -Filter 'Enabled' # Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled' # Equivalent to 'Enabled -eq $false'
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```powershell
Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle" }
Get-<Noun> -Filter { Shape -like 'C' }

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z" }
$d = [DateTime]"2010-08-23T12:30:00.0Z"
Get-<Noun> -Filter { StartTime -ge $d }
Get-<Noun> -Filter { StartTime -ge (Get-Date).AddDays(-1) }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge '-2' }  # Two days ago
Get-<Noun> -Filter { StartTime -ge '-1:30' }  # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge '-0:0:30' }  # 30 seconds ago

ARRAY PROPERTIES
When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings. Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*" }
Get-<Noun> -Filter { Users -contains "Fred*" }

You can also use the singular form with -Filter to search using other operators:

# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter { User -eq "Frederick" }
# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter { User -lt 'F' }

COMPLEX EXPRESSIONS
When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

```
Get-<Noun> -Filter { Shape -eq 'Circle' -or Shape -eq 'Square' }
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }
```

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

```
Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue' -and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue' -and Shape -eq 'Circle') }
```

### PAGING

The simplest way to page through data is to use the -Skip and -MaxRecordCount parameters. So, to read the first three pages of data with 10 records per page, use:

```
Get-<Noun> -Skip 0  -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 10 -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 20 -MaxRecordCount 10 <other filtering criteria>
```

You must include the same filtering criteria on each call, and ensure that the data is sorted consistently.

The above approach is often acceptable, but as each call performs an independent query, data changes can result in records being skipped or appearing twice. One approach to improve this is to sort by a unique id field and then start the search for the next page at the unique id after the last unique id of the previous page. For example:

```
# Get the first page
Get-<Noun> -MaxRecordCount 10 -SortBy SerialNumber

<table>
<thead>
<tr>
<th>SerialNumber</th>
</tr>
</thead>
<tbody>
<tr>
<td>A120004</td>
</tr>
<tr>
<td>A120007</td>
</tr>
<tr>
<td>... 7 other records ...</td>
</tr>
<tr>
<td>A120900</td>
</tr>
</tbody>
</table>

# Get the next page
Get-<Noun> -MaxRecordCount 10 -Filter { FirstName -gt 'A120900' }

<table>
<thead>
<tr>
<th>SerialNumber</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
```
FILTER SYNTAX DEFINITION

(Filter) ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= "{" <ComponentList> "}"

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

  <Component>

<Component> ::= <NotOperator> <Factor> |

  <Factor>

<Factor> ::= "{" <ComponentList> "}" |

  <PropertyName> <ComparisonOperator> <Value> |
  <PropertyName>

<AndOrOperator> ::= "-and" | "-or"

<NotOperator> ::= "-not" | "!

<ComparisonOperator>

  ::= "-eq" | "-ne" | "-le" | "-ge" | "-lt" | "-gt" |
  "-like" | "-notlike" | "-contains" | "-notcontains" |
  "-in" | "-notin"

(PropertyName) ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

  <scalar variable> | <array variable> |
  "$null" | "$true" | "$false"
Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30" means 1 hour and 30 minutes ago.
about_providers

Oct 10, 2013

TOPIC

about_Providers

SHORT DESCRIPTION

Describes how Windows PowerShell providers provide access to data and components that would not otherwise be easily accessible at the command line. The data is presented in a consistent format that resembles a file system drive.

LONG DESCRIPTION

Windows PowerShell providers are Microsoft .NET Framework-based programs that make the data in a specialized data store available in Windows PowerShell so that you can view and manage it.

The data that a provider exposes appears in a drive, and you access the data in a path like you would on a hard disk drive. You can use any of the built-in cmdlets that the provider supports to manage the data in the provider drive. And, you can use custom cmdlets that are designed especially for the data.

The providers can also add dynamic parameters to the built-in cmdlets. These are parameters that are available only when you use the cmdlet with the provider data.

BUILT-IN PROVIDERS

Windows PowerShell includes a set of built-in providers that you can use to access the different types of data stores.

<table>
<thead>
<tr>
<th>Provider</th>
<th>Drive</th>
<th>Data store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alias</td>
<td></td>
<td>Windows PowerShell aliases</td>
</tr>
<tr>
<td>Certificate</td>
<td>Cert:</td>
<td>x509 certificates for digital signatures</td>
</tr>
<tr>
<td>Environment</td>
<td>Env:</td>
<td>Windows environment variables</td>
</tr>
<tr>
<td>FileSystem</td>
<td>*</td>
<td>File system drives, directories, and files</td>
</tr>
<tr>
<td>Function</td>
<td></td>
<td>Windows PowerShell functions</td>
</tr>
<tr>
<td>Registry</td>
<td>HKLM:, HKCU</td>
<td>Windows registry</td>
</tr>
<tr>
<td>Variable</td>
<td>Variable:</td>
<td>Windows PowerShell variables</td>
</tr>
<tr>
<td>WS-Management</td>
<td>WS-Man</td>
<td>WS-Management configuration information</td>
</tr>
</tbody>
</table>

* The FileSystem drives vary on each system.

You can also create your own Windows PowerShell providers, and you can install providers that others develop. To list the providers that are available in your session, type:

```
get-psprovider
```
INSTALLING AND REMOVING PROVIDERS

Windows PowerShell providers are delivered to you in Windows PowerShell snap-ins, which are .NET Framework-based programs that are compiled into .dll files. The snap-ins can include providers and cmdlets.

Before you use the provider features, you have to install the snap-in and then add it to your Windows PowerShell session. For more information, see about_PsSnapins.

You cannot uninstall a provider, although you can remove the Windows PowerShell snap-in for the provider from the current session. If you do, you will remove all the contents of the snap-in, including its cmdlets.

To remove a provider from the current session, use the Remove-PsSnapin cmdlet. This cmdlet does not uninstall the provider, but it makes the provider unavailable in the session.

You can also use the Remove-PsDrive cmdlet to remove any drive from the current session. This data on the drive is not affected, but the drive is no longer available in that session.

VIEWING PROVIDERS

To view the Windows PowerShell providers on your computer, type:

get-psprovider

The output lists the built-in providers and the providers that you added to the session.

THE PROVIDER CMDLETS

The following cmdlets are designed to work with the data exposed by any provider. You can use the same cmdlets in the same way to manage the different types of data that providers expose. After you learn to manage the data of one provider, you can use the same procedures with the data from any provider.

For example, the New-Item cmdlet creates a new item. In the C: drive that is supported by the FileSystem provider, you can use New-Item to create a new file or folder. In the drives that are supported by the Registry provider, you can use New-Item to create a new registry key. In the Alias: drive, you can use New-Item to create a new alias.

For detailed information about any of the following cmdlets, type:

get-help <cmdlet-name> -detailed

CHILDITEM CMDLETS

Get-ChildItem

CONTENT CMDLETS

Add-Content
Clear-Content
Get-Content
Set-Content

ITEM CMDLETS

Clear-Item
Copy-Item
Get-Item
Invoke-Item
Move-Item
New-Item
Remove-Item
Rename-Item
Set-Item

ITEMPROPERTY CMDLETS

Clear-ItemProperty
Copy-ItemProperty
Get-ItemProperty
Move-ItemProperty
New-ItemProperty
Remove-ItemProperty
Rename-ItemProperty
Set-ItemProperty

LOCATION CMDLETS

Get-Location
Pop-Location
Push-Location
Set-Location

PATH CMDLETS

Join-Path
Convert-Path
Split-Path
Resolve-Path
Test-Path
PSDRIVE CMDLETS

Get-PSDrive
New-PSDrive
Remove-PSDrive

PSPROVIDER CMDLETS

Get-PSProvider

VIEWING PROVIDER DATA
The primary benefit of a provider is that it exposes its data in a familiar and consistent way. The model for data presentation is a file system drive.

To use data that the provider exposes, you view it, move through it, and change it as though it were data on a hard drive. Therefore, the most important information about a provider is the name of the drive that it supports.

The drive is listed in the default display of the Get-PsProvider cmdlet, but you can get information about the provider drive by using the Get-PSDrive cmdlet. For example, to get all the properties of the Function: drive, type:

get-psdrive Function | format-list *

You can view and move through the data in a provider drive just as you would on a file system drive.

To view the contents of a provider drive, use the Get-Item or Get-ChildItem cmdlets. Type the drive name followed by a colon (:). For example, to view the contents of the Alias: drive, type:

    get-item alias:

You can view and manage the data in any drive from another drive by including the drive name in the path. For example, to view the HKLM\Software registry key in the HKLM: drive from another drive, type:

    get-childitem hklm:\software

To open the drive, use the Set-Location cmdlet. Remember the colon when you specify the drive path. For example, to change your location to the root directory of the Cert: drive, type:
Then, to view the contents of the Cert: drive, type:

**get-childitem**

**MOVING THROUGH HIERARCHICAL DATA**

You can move through a provider drive just as you would a hard disk drive. If the data is arranged in a hierarchy of items within items, use a backslash (\) to indicate a child item. Use the following format:

drive:\location\child-location\...

For example, to change your location to the HKLM\Software registry key, type a Set-Location command, such as:

```powershell
set-location hklm:\software
```

You can also use relative references to locations. A dot (.) represents the current location. For example, if you are in the HKLM\Software\Microsoft registry key, and you want to list the registry subkeys in the HKLM\Software\Microsoft\PowerShell key, type the following command:

```powershell
get-childitem .\powershell
```

**FINDING DYNAMIC PARAMETERS**

Dynamic parameters are cmdlet parameters that are added to a cmdlet by a provider. These parameters are available only when the cmdlet is used with the provider that added them.

For example, the Cert: drive adds the CodeSigningCert parameter to the Get-Item and Get-ChildItem cmdlets. You can use this parameter only when you use Get-Item or Get-ChildItem in the Cert: drive.

For a list of the dynamic parameters that a provider supports, see the Help file for the provider. Type:

```powershell
get-help <provider-name>
```

For example:

```powershell
get-help certificate
```

**LEARNING ABOUT PROVIDERS**

Although all provider data appears in drives, and you use the same methods to move through them, the similarity stops there. The data stores that the provider exposes can be as varied as Active Directory locations and Microsoft Exchange Server mailboxes.
For information about individual Windows PowerShell providers, type:

get-help <ProviderName>

For example:

get-help registry

For a list of Help topics about the providers, type:

get-help * -category provider

SEE ALSO

about_Locations

about_Path_Syntax
Add-ProvSchemeControllerAddress

Oct 10, 2013
Adds a list of host names (as DNS addresses) to a provisioning scheme.

Syntax

Add-ProvSchemeControllerAddress [-ProvisioningSchemeName] <String> [-ControllerAddress] <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters]
Add-ProvSchemeControllerAddress -ProvisioningSchemeUid <Guid> [-ControllerAddress] <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters]

Detailed Description

Provides the ability to associate controller hosts (and hence implicitly a set of brokers) with a specific provisioning scheme. This optional data is passed to the machines created by the Machine Creation Services, where it is used to associate the newly created machine with a broker. The list is returned along with the provisioning scheme that it is assigned to.

Related topics

Get-ProvScheme
Remove-ProvSchemeControllerAddress

Parameters

- **-ProvisioningSchemeName**:<String>
The name for the provisioning scheme that the list of addresses is to be added to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-ProvisioningSchemeUid**:<Guid>
The unique identifier for the provisioning scheme that the list of addresses is to be added to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-ControllerAddress**:<String[]>
Specifies the array of DNS names to be added to the provisioning scheme.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-LoggingId**:<Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress**:<String>
Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>
Input Type
Citrix.MachineCreation.Sdk.ProvisioningScheme You can pipe an object containing a parameter called 'ProvisioningSchemeName' to Add-ProvSchemeControllerAddress.

Return Values
Citrix.MachineCreation.Sdk.ProvisioningScheme
Add-ProvSchemeControllerAddress returns the updated ProvisioningScheme object containing the union of the old and new controller address lists.

ProvisioningSchemeUid <Guid>
The unique identifier for the provisioning scheme.

ProvisioningSchemeName <string>
The name of the provisioning scheme.

CpuCount <int>
The number of processors that VMs will be created with when using this scheme.

MemoryMB <int>
The maximum amount of memory that VMs will be created with when using this scheme.

MasterImageVM <string>
The path within the hosting unit provider to the VM or snapshot of which the scheme is currently using a copy.

MasterImageVMDate <DateTime>
The date and time that the copy of the VM image was made for the scheme.

IdentityPoolUid <Guid>
The unique identifier of the identity pool (from the ADIdentity PowerShell snap-in) that the scheme uses.

IdentityPoolName <string>
The name of the identity pool (from the ADIdentity PowerShell snap-in) that the scheme uses.

HostingUnitUid <Guid>
The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the scheme will use.

HostingUnitName <string>
The name of the hosting unit (from the Hosting Unit PowerShell snap-in) that the scheme will use.

CleanOnBoot <Boolean>
Indicates whether or not the VMs created are to be reset to a clean state on each boot.

TaskId <Guid>
The identifier of any current task that is running for the provisioning scheme.

The metadata associated with this provisioning scheme.

ControllerAddress <string[]>
The DNS names of the controllers associated with this provisioning scheme for Quick Deploy purposes.

Notes
In the case of failure, the following errors can result.

Error Codes
-----------
ProvisioningSchemeNotFound
The specified provisioning scheme could not be located.

PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
The operation could not be performed because of a configuration logging error.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred.

To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Add-ProvSchemeControllerAddress -ProvisioningSchemeUid "01a4a008-8ce8-4165-ba9c-cdf15a6b0501" -ControllerAddress (ddcA.citrix.com,ddcB.citrix.com,ddcC.citrix2.com)

ProvisioningSchemeUid  : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
ProvisioningSchemeName : Scheme2
CpuCount               : 1
MemoryMB               : 1024
MasterImageVM          : Base.vm/Base.snapshot
MasterImageVMDate      : 17/05/2010 09:53:40
IdentityPoolUid        : 03743136-e43b-4a87-a7f4-ab71686b3c16
IdentityPoolName       : idPool1
HostingUnitUid         : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
HostingUnitName        : HostUnit1
CleanOnBoot            : True
TaskId                 : 00000000-0000-0000-0000-000000000000
Metadata               : {}
ControllerAddress      : {ddcA.citrix.com,ddcB.citrix.com,ddcC.citrix2.com}

Add a set of controllers to the provisioning scheme with the identifier "01a4a008-8ce8-4165-ba9c-cdf15a6b0501".

-------------------------- EXAMPLE 2 --------------------------

C:\PS>Get-ProvScheme -ProvisioningSchemeName scheme1 | Add-ProvSchemeControllerAddress -ControllerAddress (ddcA.citrix.com,ddcB.citrix.com,ddcC.citrix2.com)

ProvisioningSchemeUid  : 7585f0de-192e-4847-a6d8-22713c3a2942
ProvisioningSchemeName : Scheme1
CpuCount               : 1
MemoryMB               : 1024
MasterImageVM          : Base.vm/Base.snapshot
MasterImageVMDate      : 17/05/2010 09:53:40
IdentityPoolUid        : 03743136-e43b-4a87-a7f4-ab71686b3c16
IdentityPoolName       : idPool1
HostingUnitUid         : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
HostingUnitName        : HostUnit1
CleanOnBoot            : True
TaskId                 : 00000000-0000-0000-0000-000000000000
Metadata               : {}
ControllerAddress      : {ddcA.citrix.com,ddcB.citrix.com,ddcC.citrix2.com}

Add controller addresses to a provisioning scheme using a ProvisioningScheme object.
Add-ProvSchemeMetadata

Oct 10, 2013
Adds metadata on the given ProvisioningScheme.

Syntax

Add-ProvSchemeMetadata [-ProvisioningSchemeUid] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Add-ProvSchemeMetadata [-ProvisioningSchemeUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Add-ProvSchemeMetadata [-ProvisioningSchemeName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Add-ProvSchemeMetadata [-ProvisioningSchemeName] <String> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Add-ProvSchemeMetadata [-InputObject] <ProvisioningScheme> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Add-ProvSchemeMetadata [-InputObject] <ProvisioningScheme> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability for additional custom data to be stored against given ProvisioningScheme objects. This cmdlet will not overwrite existing metadata on an object - use the Set-ProvSchemeMetadata cmdlet instead.

Related topics

Set-ProvSchemeMetadata
Remove-ProvSchemeMetadata

Parameters

-ProvisioningSchemeUid <Guid>

Id of the ProvisioningScheme

| Required? | true |
| Default Value |

Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-ProvisioningSchemeName <String>

Name of the ProvisioningScheme

| Required? | true |
| Default Value |

Accept Pipeline Input? | true (ByValue, ByPropertyName) |
**-InputObject** <ProvisioningScheme[]>

Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-Name**<String>

Specifies the property name of the metadata to be added. The property must be unique for the ProvisioningScheme specified. The property cannot contain any of the following characters \/:#.*?=<>|[]()

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Value**<String>

Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Map**<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**: `<String>`

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**


Add-ProvSchemeMetadata returns an array of objects containing the new definition of the metadata.

- **Property**: `<string>`
  - Specifies the name of the property.

- **Value**: `<string>`
  - Specifies the value for the property.

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

---

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DuplicateObject

One of the specified metadata already exists.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.
AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> Add-ProvSchemeMetadata -ProvisioningSchemeUid 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the ProvisioningScheme with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Add-ProvSchemeScope

Oct 10, 2013
Add the specified ProvisioningScheme(s) to the given scope(s).

Syntax

Add-ProvSchemeScope [-Scope] <String[]> -InputObject <ProvisioningScheme[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Add-ProvSchemeScope [-Scope] <String[]> -ProvisioningSchemeUid <Guid[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Add-ProvSchemeScope [-Scope] <String[]> -ProvisioningSchemeName <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The Add-ProvSchemeScope cmdlet is used to associate one or more ProvisioningScheme objects with given scope(s).

There are multiple parameter sets for this cmdlet, allowing you to identify the ProvisioningScheme objects in different ways:

- ProvisioningScheme objects can be piped in or specified by the InputObject parameter
- The ProvisioningSchemeUid parameter specifies objects by ProvisioningSchemeUid
- The ProvisioningSchemeName parameter specifies objects by ProvisioningSchemeName (supports wildcards)

To add a ProvisioningScheme to a scope you need permission to change the scopes of the ProvisioningScheme and permission to add objects to all of the scopes you have specified.

If the ProvisioningScheme is already in a scope, that scope will be silently ignored.

Related topics

Remove-ProvSchemeScope
Get-ProvScopedObject

Parameters

- **-Scope**<String[]>

  Specifies the scopes to add the objects to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
- **InputObject</ProvisioningScheme[]>**
  Specifies the ProvisioningScheme objects to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **ProvisioningSchemeUid</Guid[]>**
  Specifies the ProvisioningScheme objects to be added by ProvisioningSchemeUid.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **ProvisioningSchemeName</String[]>**
  Specifies the ProvisioningScheme objects to be added by ProvisioningSchemeName.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **LoggingId</Guid>**
  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress</String>**
  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>
Accept Pipeline Input?  false

Return Values
None

Notes
If the command fails, the following errors can be returned.

Error Codes
---------

UnknownObject
One of the specified objects was not found.

ScopeNotFound
One of the specified scopes was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.
Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS> Add-ProvSchemeScope Finance -ProvisioningSchemeUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
Adds a single ProvisioningScheme to the 'Finance' scope.

-------------------------- EXAMPLE 2 --------------------------
c:\PS> Add-ProvSchemeScope Finance,Marketing -ProvisioningSchemeUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
Adds a single ProvisioningScheme to the multiple scopes.

-------------------------- EXAMPLE 3 --------------------------
c:\PS> Get-ProvScheme | Add-ProvSchemeScope Finance
Adds all visible ProvisioningScheme objects to the 'Finance' scope.

-------------------------- EXAMPLE 4 --------------------------
c:\PS> Add-ProvSchemeScope Finance -ProvisioningSchemeName A*
Adds ProvisioningScheme objects with a name starting with an 'A' to the 'Finance' scope.
Add-ProvTaskMetadata

Oct 10, 2013
Adds metadata on the given Task.

Syntax

Add-ProvTaskMetadata [-TaskId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Add-ProvTaskMetadata [-TaskId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]


Detailed Description

Use this cmdlet to store additional custom data against given Task objects. This cmdlet does not overwrite existing metadata on an object - use the Set-ProvTaskMetadata cmdlet instead.

Related topics

Set-ProvTaskMetadata
Remove-ProvTaskMetadata
Get-ProvTask
Stop-ProvTask
Remove-ProvTask
Switch-ProvTask

Parameters

- **TaskId**<Guid>

Id of the Task

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **InputObject**<Task[]>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.2209
Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Map<PSObject>**

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name<String>**

Specifies the property name of the metadata to be added. The property must be unique for the Task specified. The property cannot contain any of the following characters \/:#.*?=<>|[]()"'

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Value<String>**

Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId<Guid>**

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
**-AdminAddress (String)**

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**


Add-ProvTaskMetadata returns an array of objects containing the new definition of the metadata.

**Property (String)**

Specifies the name of the property.

**Value (String)**

Specifies the value for the property.

**Notes**

If the command fails, the following errors can result.

**Error Codes**

---------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DuplicateObject

One of the specified metadata already exists.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS> Add-ProvTaskMetadata -TaskId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Task with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Get-ProvDBConnection

Oct 10, 2013
Gets the database string for the specified data store used by the MachineCreation Service.

Syntax
Get-ProvDBConnection [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Returns the database connection string for the specified data store.
If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

Related topics
Get-ProvServiceStatus
Set-ProvDBConnection
Test-ProvDBConnection

Parameters
-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values
system.String
The database connection string configured for the MachineCreation Service.

Notes
If the command fails, the following errors can be returned.

Error Codes

NoDBConnections
The database connection string for the MachineCreation Service has not been specified.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ProvDBConnection

Server=serverName\SQLEXPRESS;Initial Catalog = databaseName;  Integrated Security = True

Get the database connection string for the MachineCreation Service.
Get-ProvDBSchema

Oct 10, 2013

Gets a script that creates the MachineCreation Service database schema for the specified data store.

Syntax

Get-ProvDBSchema [-DatabaseName <String>] [-ServiceGroupName <String>] [-ScriptType <ScriptTypes>] [-LocalDatabase] [-Sid <String>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Gets SQL scripts that can be used to create a new MachineCreation Service database schema, add a new MachineCreation Service to an existing site, remove a MachineCreation Service from a site, or create a database server logon for a MachineCreation Service. If no Sid parameter is provided, the scripts obtained relate to the currently selected MachineCreation Service instance, otherwise the scripts relate to MachineCreation Service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied. The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to a MachineCreation SDK cmdlet. The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured. The database scripts support only Microsoft SQL Server or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SMDCMD mode'. The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to MachineCreation Service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to MachineCreation Service roles

If ScriptType is Evict, the returned script contains:

- Removal of MachineCreation Service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

If the service uses two data stores they can exist in the same database. You do not need to configure a database before
using this command.

Related topics

Set-ProvDBConnection

Parameters

- **DatabaseName**:<String>

Specifies the name of the database for which the schema will be generated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ServiceGroupName**:<String>

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the MachineCreation services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ScriptType**:<ScriptTypes>

Specifies the type of database script returned. Available script types are:

Database

Returns a full database script that can be used to create a database schema for the MachineCreation Service in a database instance that does not already contain a schema for this service. The DatabaseName and ServiceGroupName parameters must be specified to create a script of this type.

Instance

Returns a permissions script that can be used to add further MachineCreation services to an existing database instance that already contains the full MachineCreation service schema, associating the services to the Service Group. The Sid parameter can optionally be specified to create a script of this type.

Login

Returns a database logon script that can be used to add the required logon accounts to an existing database instance that contains the MachineCreation Service schema. This is used primarily when creating a mirrored database environment. The DatabaseName parameter must be specified to create a script of this type.
Evict

Returns a script that can be used to remove the specified MachineCreation Service from the database entirely. The DatabaseName and Sid parameters must be specified to create a script of this type.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**LocalDatabase**<SwitchParameter>

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for MachineCreation services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Sid**<String>

Specifies the SID of the controller on which the MachineCreation Service instance to remove from the database is running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.String

A string containing the required SQL script for application to a database.
Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

Create the database schema.
Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned.

Error Codes
-----------

GetSchemasFailed
The database schema could not be found.

ActiveDirectoryAccountResolutionFailed
The specified Active Directory account or Group could not be found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ProvDBSchema -DatabaseName MyDB -ServiceGroupName MyServiceGroup > c:\ProvSchema.sql
Get the full database schema for site data store of the MachineCreation Service and copy it to a file called 'c:\ProvSchema.sql'.

This script can then be used to create the schema in a pre-existing database named 'MyDB' that does not already contain a MachineCreation Service site schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-ProvDBSchema -DatabaseName MyDB -scriptType Login > c:\MachineCreationLogins.sql
Get the logon scripts for the MachineCreation Service.
Get-ProvDBVersionChangeScript

Oct 10, 2013
Gets a script that updates the MachineCreation Service database schema.

Syntax

Get-ProvDBVersionChangeScript -DatabaseName <String> -TargetVersion <Version> [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

Returns a database script that can be used to upgrade or downgrade the site or secondary schema for the MachineCreation Service from the current schema version to a different version.

Related topics

Get-ProvInstalledDBVersion

Parameters

-`DatabaseName`<String>

Specifies the name of the database instance to which the update applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-`TargetVersion`<Version>

Specifies the version of the database you want to update to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-`AdminAddress`<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values

System.Management.Automation.PSObject

A PSObject containing the required SQL script for application to a database.

Notes

The PSObject returned by this cmdlet contains the following properties:

-- Script The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

-- NeedExclusiveAccess Indicates whether all services in the service group must be shut down during the update or not.

-- CanUndo Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any MachineCreation services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-ProvServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned.

Error Codes

----------

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the MachineCreation Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

------------------------ EXAMPLE 1 ------------------------

C:\PS> $update = Get-ProvDBVersionChangeScript -DatabaseName MyDb -TargetVersion 1.0.75.0
C:\PS> $update.Script > update_75.sql

Gets an SQL update script to update the current schema to version 1.0.75.0. The resulting update_75.sql script is suitable for direct use with the SQL Server SQLCMD utility.
Get-ProvInstalledDBVersion

Oct 10, 2013
Gets a list of all available database schema versions for the MachineCreation Service.

Syntax

Get-ProvInstalledDBVersion [-Upgrade] [-Downgrade] [-AdminAddress <String>] [CommonParameters>

Detailed Description

Returns the current version of the MachineCreation Service database schema, if no flags are set, otherwise returns versions for which upgrade or downgrade scripts are available and have been stored in the database.

Related topics

Parameters

- **-Upgrade**<SwitchParameter>

   Specifies that only schema versions to which the current database version can be updated should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Downgrade**<SwitchParameter>

   Specifies that only schema versions to which the current database version can be reverted should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

- **-AdminAddress**<String>

   Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Return Values
The Get-ProvInstalledDbVersion command returns objects containing the new definition of the MachineCreation Service database schema version.

Major <Integer>
Minor <Integer>
Build <Integer>
Revision <Integer>

Notes
If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination
Both the Upgrade and Downgrade flags were specified.

NoOp
The operation was successful but had no effect.

NoDBConnections
The database connection string for the MachineCreation Service has not been specified.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ProvInstalledDBVersion

Major Minor Build Revision  
----- ----- ----- --------
5    6    0    0         

Get the currently installed version of the MachineCreation Service database schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-ProvInstalledDBVersion -Upgrade

Major Minor Build Revision  
----- ----- ----- --------
6    0    0    0         

Get the versions of the MachineCreation Service database schema for which upgrade scripts are supplied.
Get-ProvObjectReference

Oct 10, 2013
Returns the number of local objects holding references to objects from other services.

Syntax

Get-ProvObjectReference [-HostingUnitUid <Guid[]>] [-IdentityPoolUid <Guid[]>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns for each hosting unit or identity pool GUID the number of references by provisioning schemes or by long running task. This check is done without regard for scoping of existing provisioning schemes, references by inaccessible schemes are also checked.

Related topics

Parameters

-HostingUnitUid<Guid[]>

The identifiers of the hosting units(s) to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-IdentityPoolUid<Guid[]>

The identifiers of the identity pool(s) to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values

ObjectReferenceCount

An object which contain the input object identifier, its type, the type of referencing object and the number of references.

Notes

In the case of failure, the following errors can result.

Error Codes

----------

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

------------------------------- EXAMPLE 1 -------------------------------

Get-ProvObjectReference -HostingUnitUid 5B66A060-85E1-4DBD-9D1B-BF79881D3BB1

<table>
<thead>
<tr>
<th>Count</th>
<th>ObjectId</th>
<th>Source</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5b66a060-85e1-4dbd-9d1b-bf79881d3bb1</td>
<td>ProvisioningScheme</td>
<td>HostingUnit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Count</th>
<th>ObjectId</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5b66a060-85e1-4dbd-9d1b-bf79881d3bb1</td>
<td>Task</td>
</tr>
</tbody>
</table>
Target : HostingUnit
This checks a single hosting unit for objects that have references to them; in this case we only have a single provisioning scheme that relates to it.
Get-ProvScheme

Oct 10, 2013

Gets the list of provisioning schemes.

Syntax


Detailed Description

Lets you retrieve the list of defined provisioning schemes.

Related topics

New-ProvScheme
Remove-ProvScheme
Add-ProvSchemeMetadata
Remove-ProvSchemeMetadata
Add-ProvSchemeControllerAddress
Remove-ProvSchemeControllerAddress

Parameters

-ProvisioningSchemeName <String>

The name of the provisioning scheme.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ScopeId <Guid>
Gets only results with a scope matching the specified scope identifier.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ScopeName</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ScopeName**<String>

Gets only results with a scope matching the specified scope name.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ScopeName</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **ReturnTotalRecordCount**<SwitchParameter>

See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ReturnTotalRecordCount</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **MaxRecordCount**<Int32>

See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-MaxRecordCount</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Skip**<Int32>

See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Skip</td>
<td>false</td>
<td>0</td>
<td>false</td>
</tr>
</tbody>
</table>

- **SortBy**<String>
See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter<String>**

See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. When a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.MachineCreation.Sdk.ProvisioningScheme

This object provides details of the provisioning scheme and contains the following information:

**ProvisioningSchemeUId <Guid>**

The unique identifier for the provisioning scheme.

**ProvisioningSchemeName <string>**

The name of the provisioning scheme.

**CpuCount <int>**

The number of processors that VMs will be created with when using this scheme.

**MemoryMB <int>**

The maximum amount of memory that VMs will be created with when using this scheme.

**MasterImageVM <string>**
The path within the hosting unit provider to the copy of the VM snapshot that the scheme uses.

MasterImageVMDate <DateTime>

The date and time that the copy was made of the VM snapshot used by the scheme.

IdentityPoolUid <Guid>

The unique identifier of the identity pool (from the ADIdentity PowerShell snap-in) that the scheme uses.

IdentityPoolName <string>

The name of the identity pool (from the ADIdentity PowerShell snap-in) that the scheme uses.

HostingUnitUid <Guid>

The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the scheme uses.

HostingUnitName <string>

The name of the hosting unit (from the Hosting Unit PowerShell snap-in) that the scheme uses.

CleanOnBoot <Boolean>

Indicates whether the VMs that are created will be reset to a clean state on each boot.

TaskId <Guid>

The identifier of any current task that is running for the provisioning scheme.


The metadata associated with this provisioning scheme.

ControllerAddress <string[]>

The DNS names of the controllers associated with this provisioning scheme for Quick Deploy purposes.

VMMetadata <char[]>

The opaque VM metadata block.

UsePersonalVDiskStorage <bool>

True if the scheme will use personal vDisk storage.

PersonalVDiskDriveLetter <char>

The drive letter for the personal vDisk.

PersonalVDiskDriveSize <int>

The size of the personal vDisk in GB.

ProfileUsagePercentage <double>
The percentage of the personal vDisk to be used for profile data

Notes
In the case of failure, the following errors can result.

Error Codes

----------

PartialData
Only a subset of the available data was returned.

CouldNotQueryDatabase
The query to get the database was not defined.

PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error

CommunicationError
An error occurred while communicating with the service.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Get-ProvScheme

ProvisioningSchemeUid : 7585f0de-192e-4847-a6d8-22713c3a2f42
ProvisioningSchemeName : Scheme1
CpuCount : 1
MemoryMB : 1024
MasterImageVM : /Base.vm/base.snapshot
Returns all of the available provisioning schemes.

-------------------------- EXAMPLE 2 --------------------------

C:\PS>Get-ProvScheme -ProvisioningSchemeName Scheme[0-1]

ProvisioningSchemeUid : 7585f0de-192e-4847-a6d8-22713c3a2f42
ProvisioningSchemeName : Scheme1
CpuCount : 1
MemoryMB : 1024
MasterImageVM : /Base.vm/base.snapshot
MasterImageVMDate : 17/05/2010 09:27:50
IdentityPoolUid : 03743136-e43b-4a87-af74-ab71686b3c16
IdentityPoolName : idPool1
HostingUnitUid : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
HostingUnitName : HostUnit1
CleanOnBoot : True
TaskId : 00000000-0000-0000-0000-000000000000
Metadata : {}
ControllerAddress : {}
VMMetadata : {0, 1, 0, 0...}

Returns all of the provisioning schemes that have the name 'Scheme0' or 'Scheme1'.
Get-ProvSchemeMasterVMImageHistory

Oct 10, 2013

Gets the list of master VM snapshots that have been used to provide hard disks to provisioning schemes.

Syntax


Detailed Description

Provides the ability to discover the master VM snapshots that have been used previously to provide the hard disk image for provisioning schemes. This information includes the date and time when the image was introduced. This information can be used to roll back a provisioning scheme to a previous image.

Related topics

Publish-ProvMasterVMImage

Remove-ProvSchemeMasterVMImageHistory

Parameters

-ProvisioningSchemeName <String>

The name of the provisioning scheme.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-MasterImageVM <String>

The path to the snapshot item in the hosting unit PowerShell provider.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-VMImageHistoryUid <Guid>

The unique identifier for the Image History item.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-ReturnTotalRecordCount <SwitchParameter>
See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMImageHistoryUid</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-MaxRecordCount\<Int32> 
See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-Skip\<Int32> 
See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>0</td>
<td>false</td>
</tr>
</tbody>
</table>

-SortBy\<String> 
See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-Filter\<String> 
See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress\<String> 
Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
Citrix.MachineCreation.Sdk.VMImage
The object that represents a master VM history. This contains the following parameters:

VMImageHistoryUid \<Guid>
The unique identifier for the History item.

ProvisioningSchemeUid <Guid>

The unique identifier for the provisioning scheme that the VM was used for.

ProvisioningSchemeName <string>

The name of the provisioning scheme that the VM was used for.

MasterImageVM <string>

The path to the Snapshot item that was used as the master VM image.

Date <DateTime>

The date and time that the VM or snapshot was used in the provisioning scheme.

Notes

In the case of failure, the following errors can result.

Error Codes

-----------

PartialData

Only a subset of the available data was returned.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CouldNotQueryDatabase

The query required to get the database was not defined.

CommunicationError

An error occurred while communicating with the service.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Get-ProvSchemeMasterVMImageHistory

VMImageHistoryUid : 3cba3a75-89cd-4868-989b-27feb378fec5
ProvisioningSchemeUid : 7585f0de-192e-4847-a6d8-22713c3a2f42
ProvisioningSchemeName : MyScheme
MasterImageVM : /Base.vm/base.snapshot
Date : 17/05/2010 09:27:50

Gets all the hard disk images that have been used across all provisioning schemes.
-------------------------- EXAMPLE 2 --------------------------

C:\PS>Get-ProvSchemeMasterVMI mageHistory -ProvisioningScheme MyScheme -masterImageVM "\BaseXp.vm\update1.snapshot" | Publish-ProvMasterVmImage
Rollback the provisioning scheme to use the hard disk from the update1.snapshot for the provisioning scheme called "MyScheme".
Get-ProvScopedObject

Oct 10, 2013
Gets the details of the scoped objects for the MachineCreation Service.

Syntax


Detailed Description

Returns a list of directly scoped objects including the names and identifiers of both the scope and object as well as the object description for display purposes.

There will be at least one result for every directly scoped object. When an object is associated with multiple scopes the output contains one result per scope duplicating the object details.

No records are returned for the All scope, though if an object is not in any scope a result with a null ScopeId and ScopeName will be returned.

Related topics

Parameters

- **-ScopeId**<Guid>

  Gets scoped object entries for the given scope identifier.

  | Required? | false |
  | Default Value |
  | Accept Pipeline Input? | true (ByPropertyName) |

- **-ScopeName**<String>

  Gets scoped object entries with the given scope name.

  | Required? | false |
  | Default Value |
  | Accept Pipeline Input? | true (ByPropertyName) |

- **-ObjectType**<ScopedObjectType>

  Gets scoped object entries for objects of the given type.

  | Required? | false |
  | Default Value |
  | Accept Pipeline Input? | false |
### -ObjectId <String>

Gets scoped object entries for objects with the specified object identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>true (ByPropertyName)</th>
</tr>
</thead>
</table>

### -ObjectName <String>

Gets scoped object entries for objects with the specified object identifier.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>true (ByPropertyName)</th>
</tr>
</thead>
</table>

### -Description <String>

Gets scoped object entries for objects with the specified description.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>false</th>
</tr>
</thead>
</table>

### -Property <String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Default Value</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Accept Pipeline Input?</th>
<th>false</th>
</tr>
</thead>
</table>

### -ReturnTotalRecordCount <SwitchParameter>

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Prov_Filtering for details.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-MaxRecordCount**<Int32>

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Skip**<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by `-ReturnTotalRecordCount`.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SortBy**<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter**<String>

Gets records that match a PowerShell-style filter expression. See `about_Prov_Filtering` for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.MachineCreation.Sdk.ScopedObject

The Get-ProvScopedObject command returns an object containing the following properties:

ScopeId <Guid?>

Specifies the unique identifier of the scope.

ScopeName <String>

Specifies the display name of the scope.

ObjectType <ScopedObjectType>

Type of the object this entry relates to.

ObjectId <String>

Unique identifier of the object.

ObjectName <String>

Display name of the object

Description <String>

Description of the object (possibly $null if the object type does not have a description).

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

UnknownObject

One of the specified objects was not found.

PartialData
Only a subset of the available data was returned.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase
The query required to get the database was not defined.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS> Get-ProvScopedObject -ObjectType Scheme
Scopeld    : eff6f464-f1ee-4442-add3-99982e0cec01
ScopeName  : Sales
ObjectType : Scheme
ObjectId   : cd4174ee-9e4b-4e57-b126-9dbf757fe493
ObjectName : MyExampleScheme
Description : Test scheme
Scopeld    : 304e0fa7-d390-47f0-a94f-7e956a324c41
ScopeName  : Finance
ObjectType : Scheme
ObjectId : cd4174ee-9e4b-4e57-b126-9dbf757fe493
ObjectName : MyExampleScheme
Description : Test scheme

ScopeId :
ScopeName :

ObjectType : Scheme
ObjectId : 5062e46b-71bc-4ac9-901a-30fe6797e2f6
ObjectName : AnotherScheme
Description : Another scheme in no scopes

Gets all of the scoped objects with type Scheme. The example output shows a scheme object (MyExampleScheme) in two scopes Sales and Finance, and another scheme (AnotherScheme) that is not in any scope. The ScopeId and ScopeName values returned are null in the final record.
Get-ProvService

Oct 10, 2013
Gets the service record entries for the MachineCreation Service.

Syntax

Detailed Description
Returns instances of the MachineCreation Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database. A database connection for the service is required to use this command.

Related topics
Parameters
-Property <String[]>
Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ReturnTotalRecordCount <SwitchParameter>
When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-MaxRecordCount <Int32>
Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
</tbody>
</table>
### Accept Pipeline Input?

| Accept Pipeline Input? | false |

### -Skip<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |

### -SortBy<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |

### -Filter<String>

Gets records that match a PowerShell-style filter expression. See about_Prov_Filtering for details.

| Required? | false |

### -AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |

### Return Values

Citrix.MachineCreation.Sdk.Service
The Get-ProvServiceInstance command returns an object containing the following properties.

Uid <Integer>
Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>
Specifies the unique identifier for the service instance.

DNSName <String>
Specifies the domain name of the host on which the service runs.

MachineName <String>
Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>
Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>
Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>
Specifies the date and time at which the service was last stopped or restarted.

OSType
Specifies the operating system installed on the host on which the service runs.

OSVersion
Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion
Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>
Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>
Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>
Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.
Notes
If the command fails, the following errors can be returned.

Error Codes
----------

UnknownObject
One of the specified objects was not found.

PartialData
Only a subset of the available data was returned.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase
The query required to get the database was not defined.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-ProvService

Uid : 1
ServiceHostId : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName : MyServer.company.com
MachineName : MYSERVER
CurrentState : On
LastStartTime : 04/04/2011 15:25:38
LastActivityTime : 04/04/2011 15:33:39
OSType : Win32NT
OSVersion : 6.1.7600.0
ServiceVersion : 5.1.0.0
DatabaseUserName : NT AUTHORITY\NETWORK SERVICE
SID : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : [MySiteService1, MySiteService2...]

Get all the instances of the MachineCreation Service running in the current service group.
Get-ProvServiceAddedCapability

Oct 10, 2013
Gets any added capabilities for the MachineCreation Service on the controller.

Syntax
Get-ProvServiceAddedCapability [-AdminAddress <String>] [CommonParameters]

Detailed Description
Enables updates to the MachineCreation Service on the controller to be detected.
You do not need to configure a database connection before using this command.

Related topics
Parameters
-CoreAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
System.String
String containing added capabilities.

Notes
If the command fails, the following errors can be returned.

Error Codes
-------
DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ProvServiceAddedCapability
Get the added capabilities of the MachineCreation Service.
Get-ProvServiceConfigurationData

Oct 10, 2013
Gets configuration data for the service.

Syntax

Detailed Description
Provides the ability to determine the configuration parameters for the service

Related topics
Set-ProvServiceConfigurationData
Remove-ProvServiceConfigurationData

Parameters
-Name <String[]>
The Configuration data item Name.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Value <String[]>
The Configuration data item value.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ReturnTotalRecordCount <SwitchParameter>
When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td>false</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------</td>
</tr>
</tbody>
</table>

**-MaxRecordCount**<Int32>

Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Skip**<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by `-ReturnTotalRecordCount`.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-SortBy**<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Filter**<String>

Gets records that match a PowerShell-style filter expression. See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.
Required? | false
---|---
Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default.
Accept Pipeline Input? | false

Return Values

Citrix.MachineCreation.Sdk.ServiceConfigurationData

This object provides details of the configuration data and contains the following information:

Name <string>
The Name for the configuration item.

Value <string>
The value for the configuration item.

Notes
In the case of failure the following errors can be produced.

Error Codes

-----------

PartialData
Only a subset of the available data was returned.

CouldNotQueryDatabase
The query required to get the database was not defined.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Get-ProvConfigurationData

Name : DeltaDiskDelete.timeDelay
Value : 10
Get the Configuration data for the service.
Get-ProvServiceInstanceId

Oct 10, 2013
Gets the service instance entries for the MachineCreation Service.

Syntax

Get-ProvServiceInstanceId [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns service interfaces published by the instance of the MachineCreation Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Related topics

Parameters

- `AdminAddress <String>`

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.MachineCreation.Sdk.ServiceInstance

The Get-ProvServiceInstanceId command returns an object containing the following properties.

ServiceGroupId <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.
ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Prov.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

- SDK - for PowerShell operations
- InterService - for operations between different services
- Peer - for communications between services of the same type


The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

--------

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:PS>Get-ProvServiceInstance

Address            : http://MyServer.com:80/Citrix/MachineCreationService
Binding            : wcf_HTTP_kerb
InterfaceType      : SDK
Metadata           :
MetadataMap        :
ServiceAccount     : ENG\MyAccount$
ServiceAccountSid  : S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName   : MyServiceGroup
ServiceGroupUid    : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceUid : 00000000-0000-0000-0000-000000000000
ServiceType        : Prov
Version            : 1

Address            : http://MyServer.com:80/Citrix/MachineCreationService/IServiceApi
Binding            : wcf_HTTP_kerb
InterfaceType      : InterService
Metadata           :
MetadataMap        :
ServiceAccount     : ENG\MyAccount
ServiceAccountSid  : S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName : MyServiceGroup
ServiceGroupUid : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceUid : 00000000-0000-0000-0000-000000000000
ServiceType : Prov
Version : 1

Get all instances of the MachineCreation Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.
Get-ProvServiceStatus

Oct 10, 2013

Gets the current status of the MachineCreation Service on the controller.

Syntax

Get-ProvServiceStatus [-AdminAddress <String>] [CommonParameters]

Detailed Description

Enables the status of the MachineCreation Service on the controller to be monitored. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured.

Related topics

Set-ProvDBConnection
Test-ProvDBConnection
Get-ProvDBConnection
Get-ProvDBSchema

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

Return Values


The Get-ProvServiceStatus command returns an object containing the status of the MachineCreation Service together with extra diagnostics information.

DBUnconfigured

The MachineCreation Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the MachineCreation Service. This may be because the service attempted
to log on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**

The expected stored procedures are missing from the database. This may be because the MachineCreation Service schema has not been added to the database.

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**

The version of the MachineCreation Service currently in use is incompatible with the version of the MachineCreation Service schema on the database. Upgrade the MachineCreation Service to a more recent version.

**DBOlderVersionThanService**

The version of the MachineCreation Service schema on the database is incompatible with the version of the MachineCreation Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**

A database schema upgrade is currently in progress.

**OK**

The MachineCreation Service is running and is connected to a database containing a valid schema.

**Failed**

The MachineCreation Service has failed.

**Unknown**

(0) The service status cannot be determined.

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

----------

**DatabaseError**

An error occurred in the service while attempting a database operation.

**DatabaseNotConfigured**

The operation could not be completed because the database for the service is not configured.

**DataStoreException**

An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------- EXAMPLE 1 ---------------
c:\PS>Get-ProvServiceStatus

DBUnconfigured
Get the current status of the MachineCreation Service.
Get-ProvTask

Oct 10, 2013
Gets the task history for the MachineCreation Service.

Syntax

Detailed Description
Returns a list of tasks that have run or are currently running within the MachineCreation Service.

Related topics
Remove-ProvTask
Stop-ProvTask
Switch-ProvTask
Add-ProvTaskMetadata
Remove-ProvTaskMetadata

Parameters

-TaskId <Guid>
Specifies the task identifier to be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-Type <JobType>
Specifies the type of task to be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-Active <Boolean>
Specifies whether currently running tasks only or completed tasks only are returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Parameter</td>
<td>Required?</td>
<td>Default Value</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------</td>
<td>------------------------</td>
</tr>
<tr>
<td>-ReturnTotalRecordCount</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>-MaxRecordCount</td>
<td>false</td>
<td>250</td>
</tr>
<tr>
<td>-Skip</td>
<td>false</td>
<td>0</td>
</tr>
<tr>
<td>-SortBy</td>
<td>false</td>
<td>The default sort order is by name or unique identifier.</td>
</tr>
</tbody>
</table>

**-ReturnTotalRecordCount** `<SwitchParameter>`

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Prov_Filtering for details.

**-MaxRecordCount** `<Int32>`

Specifies the maximum number of records to return.

**-Skip** `<Int32>`

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

**-SortBy** `<String>`

 Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.
**-Filter**<String>

Gets records that match a PowerShell-style filter expression. See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can result.

**Error Codes**

----------

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ProvTask -Active $false

TaskId                   : cfe5b3a2-c471-443e-b05e-8658e672d10f
Type                     : MyTask
Host                     : NYSERVER
Status                   : Finished
CurrentOperation         :
TaskProgress             : 100
TaskExpectedCompletion   : 10/10/2012 15:28:12
LastUpdateTime           : 10/10/2012 15:28:12
ActiveElapsedTime        : 56
DateFinished             : 10/10/2012 15:28:12
TerminatingError         :
...

Get all completed tasks for the MachineCreation Service.

All tasks will publish at least the fields listed above, plus more related to the particular task being performed.
Get-ProvVM

Oct 10, 2013

Gets the VMs that were created using Citrix XenDesktop Machine Creation Services.

Syntax


Detailed Description

Provides the ability to obtain a list of the VMs that were created using Machine Creation Services.

Related topics

New-ProvVM
Remove-ProvVM
Lock-ProvVM
Unlock-ProvVM

Parameters

-ProvisioningSchemeName <String>

The name of the provisioning scheme.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

-ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

-VMName <String>

The name of the VM in the hypervisor context.

<p>| Required? |
| Default Value |
| Accept Pipeline Input? |</p>
<table>
<thead>
<tr>
<th><strong>-Locked</strong>&lt;Boolean&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indicates whether only VMs that are marked as locked are returned or not (see Lock-ProvVM and Unlock-ProvVM for details).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-Tag</strong>&lt;String&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>The tag string that was associated with the VM when it was locked.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-OutOfDate</strong>&lt;SwitchParameter&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indicates that the image currently assigned to the VM is out of date. The image will be updated the next time the VM is restarted.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-ReturnTotalRecordCount</strong>&lt;SwitchParameter&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>See about_Prov_Filtering for details.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>-MaxRecordCount</strong>&lt;Int32&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>
See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Skip<Int32>

See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>0</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-SortBy<String>

See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-Filter<String>

See about_Prov_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values


The object has the following properties:

ADAccountSid <string>
The SID of the AD computer account that the VM is using.

ADAccountName <string>
The name of the AD computer account that the VM is using.

CpuCount <int>
The number of processors that the VM has been allocated.

CreationDate <DateTime>
The date and time that the VM was created.

Domain <string>
The Domain of the AD computer account that the VM is using.

ImageOutOfDate <Boolean>
Indicates if the image will be updated next time the VM is started.

Lock <Boolean>
Indicates whether the VM is locked or not.

MemoryMB <int>
The maximum amount of memory that VMs will be created with when using this scheme.

ProvisioningSchemeName <string>
The name of the provisioning scheme that the VM is part of.

ProvisioningSchemeUid <Guid>
The unique identifier for the provisioning scheme that the VM is part of.

Tag <string>
Provides the string associated with a locked VM.

VMId <string>
The identifier for the VM (hypervisor context).

VMName <string>
The name of the VM (hypervisor context).

AssignedImage <string>

The identifier (in the hypervisor) for the hard disk image that the VM is currently assigned.

BootedImage <string>

The identifier (in the hypervisor) for the hard disk image with which the VM is currently started.

HostingUnitUid <Guid>

The unique identifier for the hosting unit that was used to create the VM.

HypervisorConnectionUid <Guid>

The unique identifier of the hypervisor connection that was used to create the VM.

LastBootTime <DateTime>

The date and time of the last start of the VM.

OSDiskIndex <int>

The disk index at which the hard disk image, from which the VM is currently started, is attached (or [int]:MinValue for VMs inherited from versions of XenDesktop before 5.6)

PersonalVDiskIndex <int>

The disk index at which the personal vdisk is attached (defaults to [int]:MinValue for VMs without a personal vdisk)

PersonalVDiskStorage <string>

The identifier (in the hypervisor) for the storage on which the personal virtual disk image from which the VM is currently started is located. This is set only if the VM has a personal vDisk attached.

StorageId <string>

The identifier (in the hypervisor) for the storage on which the hard disk image, from which the VM is currently started, is located.

Notes

In the case of failure, the following errors can result.

Error Codes

-------------

PartialData

Only a subset of the available data was returned.

CouldNotQueryDatabase

The query required to get the database was not defined.
PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

An error occurred while communicating with the service.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Get-ProvVM -provisioningSchemeName MyScheme

ADAccountName            : MYDomain\computer2$
ADAccountSid            : S-1-5-21-3751941309-1176885247-1409628468-3179
CpuCount                : 1
CreationDate            : 17/05/2012 09:35:22
Domain                  : steve.dum.local
ImageOutOfDate          : False
Lock                    : True
MemoryMB                : 512
ProvisioningSchemeName  : XenPS
ProvisioningSchemeUid   : 5135a865-ba49-4e5f-87f2-2d65ee7a4e51
Tag                     : Brokered
VMId                    : a830de93-ddc5-b763-dc1a-35580a31401c
VMName                  : IP0051
AssignedImage           : 57fc60c3-eb9b-4d38-8646-0afceec85335
BootedImage             : 57fc60c3-eb9b-4d38-8646-0afceec85335
HostingUnitUid          : ea17840f-cf2d-4d80-94e0-3b752b32e0af
HypervisorConnectionUid : 99f9f826-31fc-4453-6ca0-9ba54306c3ac
IdentityDiskIndex       : 1
LastBootTime            : 17/05/2012 09:35:22
OSDiskIndex             : 0
PersonalVDiskIndex      : -2147483648
PersonalVDiskStorage :
Storageld : 33ad07a7-edd7-589b-716a-86cad4739f5e
Gets all the Virtual Machines that were provisioned into the Provisioning Scheme called 'MyScheme'.

-------------------------- EXAMPLE 2 --------------------------
C:\PS>Get-ProvVM -Locked $true

ADAccountName : MYDomain\computer1$
ADAccountSid : S-1-5-21-3751941309-1176885247-1409628468-3178
CpuCount : 1
CreationDate : 17/05/2012 09:35:30
Domain : steve.dum.local
ImageOutOfDate : False
Lock : True
MemoryMB : 512
ProvisioningSchemeName : XenPS
ProvisioningSchemeUid : 5135a865-ba49-4e5f-87f2-2d65ee7a4e51
Tag : Brokered
VMId : a830de93-ddc5-b763-dc1a-35580a31401c
VMName : IP0051
AssignedImage : 57fc60c3-eb9b-4d38-8646-0afceec85335
BootedImage : 57fc60c3-eb9b-4d38-8646-0afceec85335
HostingUnitUid : ea17840f-cf2d-4d80-94e0-3b752b32e0af
HypervisorConnectionUid : 999f826-31fc-4453-8ca0-9ba54306c3ac
IdentityDiskIndex : 1
LastBootTime : 17/05/2012 09:35:22
OSDiskIndex : 0
PersonalVDiskIndex : -2147483648
PersonalVDiskStorage :
Storageld : 33ad07a7-edd7-589b-716a-86cad4739f5e
Gets all the Virtual Machines that were locked, regardless of which Provisioning Scheme the VM is part of.
Lock-ProvVM

October 10, 2013
Locks a VM.

Syntax

Lock-ProvVM [-VMID] <String[]> -ProvisioningSchemeName <String> [-Tag <String>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Lock-ProvVM [-VMID] <String[]> -ProvisioningSchemeUid <Guid> [-Tag <String>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to 'lock' a virtual machine with a tag string. This stops other commands from being able to remove the virtual machine without unlocking the VM first. This is to enable consumers of the virtual machines to indicate that the VM is being used.

Related topics

UnLock-ProvVM
Get-ProvVM
Remove-ProvVM

Parameters

-VMID <String[]>
The virtual machine Id (hypervisor context).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-ProvisioningSchemeName <String>
The name of the provisioning scheme.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-ProvisioningSchemeUid <Guid>
The unique identifier of the provisioning scheme.
-Tag <String>

The string to be held against the VM being locked.

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-AdminAddress <String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

Input Type

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine You can pipe an object containing a parameter called ‘VMId’ and ‘ProvisioningSchemeName’ to Lock-ProvVM

Notes

In the case of failure, the following errors can result.

Error Codes

-----------

VMDoesNotExist
The specified VM cannot be located.

VMAlreadyLocked

The VM is already unlocked.

VMDoesNotExistForProvisioningScheme

The specified VM does exist in the hypervisor, but is not part of the specified provisioning scheme.

PermissionDenied

The user is not authorized to perform this operation

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Lock-ProvVM -provisioningSchemeName MyScheme -VMId bc79802c-ba6e-8de8-99e9-4c35d7ad24b4 -Tag LockedString

Locks the VM with the Id 'bc79802c-ba6e-8de8-99e9-4c35d7ad24b4' in the provisioning scheme 'MyScheme' with the tag 'LockedString'.

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Get-ProvVM -provisioningSchemeName MyScheme | Lock-ProvVM -Tag LockedString

Locks all the VMs in the provisioning scheme 'MyScheme' with the tag 'LockedString'.
New-ProvScheme

Oct 10, 2013
Creates a new provisioning scheme.

Syntax


Detailed Description
Lets you create a new provisioning scheme. The creation process makes a copy of the hard disk attached to a virtual machine snapshot and stores it in every storage location that the hosting unit referenced by the provisioning scheme defines. This is a long-running task and typically takes several minutes to complete (depending on the size of the hard disk that is being copied and the number of snapshots that the hard disk consists of).

A snapshot must be used rather than a VM, so that the content of the hard disk for the provisioning scheme can be easily determined.

As the snapshot is specified using a path into the Citrix Host Service Powershell Provider the Citrix Host Service Powershell snap-in must also be loaded for this cmdlet to be used.

This cmdlet requires information to be provided that is retrieved using other snap-ins that form part of the Citrix Machine Creation Services: Hosting Unit Service Snapin The snap-in that provides information about the hypervisors. AD Identity Service Snapin The snap-in that provides information about the identity pools.

The provisioning scheme is a collection of all of the data that is required to form a template against which virtual machines can be created. It therefore requires the following: Hosting Unit A reference to an item defined in the Host Service that defines the hypervisor, the network, and the storage to be used. Identity Pool A reference to the collection of Active Directory accounts that is used for virtual machines created from the provisioning scheme.

Related topics
Get-ProvTask
Get-ProvScheme
Test-ProvSchemeNameAvailable

Parameters
-ProvisioningSchemeName <String>
The name of the provisioning scheme to be created. This must be a name that is not being used by an existing provisioning scheme, and it must not contain any of the following characters /;:*?<=<>|[]()"'

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

-HostingUnitUid <Guid>
The identifier for the hosting unit used for the provisioning scheme.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

-IdentityPoolUid <Guid>
The identifier of the identity pool used for the provisioning scheme.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

-HostingUnitName <String>
The name of the hosting unit used for the provisioning scheme.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>
- **IdentityPoolName**: `<String>`
  The name for the identity pool used for the provisioning scheme.
  - Required? true
  - Default Value
  - Accept Pipeline Input? false

- **MasterImageVM**: `<String>`
  The path in the hosting unit provider to the virtual machine snapshot that is used. This identifies the hard disk to be used and the default values for the memory and processors. This must be a path to a Snapshot item in the same hosting unit that the hosting unit name or hosting unit UID refers to.
  Valid paths are of the format; XDHyp:\HostingUnits\<HostingUnitName>\<path>\<VmName>.vm\<SnapshotName>.snapshot
  - Required? true
  - Default Value
  - Accept Pipeline Input? true (ByPropertyName)

- **VMCPUCount**: `<Int32>`
  The number of processors used by virtual machines created from the provisioning scheme.
  - Required? false
  - Default Value The number of CPUs assigned to the base image VM snapshot.
  - Accept Pipeline Input? false

- **VMMemoryMB**: `<Int32>`
  The maximum amount of memory used by virtual machines created from the provisioning scheme.
  - Required? false
  - Default Value The amount of memory assigned to the base image VM snapshot.
  - Accept Pipeline Input? false

- **CleanOnBoot**: `<SwitchParameter>`
  Indicates whether or not the virtual machines created from this provisioning scheme are reset to their initial condition each time they are started.
  - Required? false
  - Default Value false
  - Accept Pipeline Input? false

- **UsePersonalVDiskStorage**: `<SwitchParameter>`
  Indicates whether or not personal vDisks are used for the VMs in this provisioning scheme.
  - Required? false
  - Default Value false
  - Accept Pipeline Input? false

- **Scope**: `<String[]>`
  The administration scopes to be applied to the new provisioning scheme.
  - Required? false
  - Default Value
  - Accept Pipeline Input? false

- **NoImagePreparation**: `<SwitchParameter>`
  Indicates that Image Preparation should not be performed on this Provisioning Scheme
  - Required? false
  - Default Value false
  - Accept Pipeline Input? false
- **NetworkMapping**<br>
Specifies how the attached NICs are mapped to networks. If this parameter is omitted, provisioned VMs are created with a single NIC, which is mapped to the default network in the HostingUnit. If this parameter is supplied, Machines are created with the number of NICs specified in the map, and each NIC is attached to the specified network.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Metadata**<br>
Specifies any metadata to be attached to the scheme when it is created.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **ResetAdministratorPasswords**<br>
Indicates whether to reset the password for the administrator accounts on provisioned machines.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **RunAsynchronously**<br>
Indicates whether or not the command returns before it is complete. If specified, the command returns an identifier for the task that was created. This task can be monitored using the get-ProvTask command.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **PurgeJobOnSuccess**<br>
Indicates that the task history is removed from the database when the task has finished. This can only be specified for tasks that are not run asynchronously.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**<br>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<br>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>localhost</td>
<td>false</td>
</tr>
</tbody>
</table>

---

**Return Values**

System.Guid
When the RunAsynchronously identifier is specified, this GUID is returned and provides the task identifier:

System.Management.Automation.PSCustomObject

This object provides details of the task that was run and contains the following information:

TaskId <Guid>
The identifier for the task that was performed.

Active <Boolean>
Indicates whether the task is still processing or is complete.

Host <string>
The name of the host on which the task is running or was run.

DateStarted <DateTime>
The date and time that the task was initiated.

Type <Citrix.XDInterServiceTypes.JobType>
The type of task. For new provisioning scheme tasks, this is always NewProvisioningScheme.

The list of metadata stored against the task. For new tasks, this is empty until metadata is added.

Indicates the status of the workflow that is used to process the task.

ProvisioningSchemeName <string>
The name of the provisioning scheme that the task was for.

ProvisioningSchemeUid <Guid>
The unique identifier for the provisioning scheme that the task was for.

MasterImage <string>
The path (in the hosting unit provider) of the virtual machine snapshot that was used as the master VM image for the task.

IdentityPoolName <string>
The name of the identity pool (from the ADIdentity PowerShell snap-in) that the new provisioning scheme uses.

IdentityPoolUid <guid>
The unique identifier name of the identity pool (from the ADIdentity PowerShell snap-in) that the new provisioning scheme uses.

HostingUnitName <string>
The name of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme uses.

HostingUnitUid

The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme uses.

PersonalVDiskDriveLetter

The drive letter on which a personal vDisk is mounted (blank if the personal vDisk feature was not selected).

PersonalVDiskDriveSize

The size of any personal vDisk (zero if the personal vDisk feature was not selected).

Scopes <Citrix.Fma.Sdk.ServiceCoreScopeReference[]>
The delegated administration scopes to which the scheme will belong.

NetworkMap <Citrix.MachineCreation.Sdk.NetworkMap>
The list of NIC to network associations, if specified.

ProvisioningSchemeMetadata <Dictionary<string, string>>
The metadata to apply to the provisioning scheme, if specified.

TaskState <Citrix.MachineCreation.Sdk.NewProvisioningSchemeState>
The state of the task. This can be any of the following:

Processing

The task has begun but has not done anything yet.

LocatingResources,
The workflow is locating resources from other services.

HostingUnitNotFound
The task failed because the required hosting unit could not be located.

VirtualMachineSnapshotNotFound
The task failed because the required VM snapshot could not be located.

ConsolidatingMasterImage
The task is consolidating the master image.

ReplicatingConsolidatedImageToAllStorage
The task is replicating the consolidated master image.

StoringProvisioningScheme
The task is storing the provisioning scheme data in the database.

Finished
The task completed with no errors.

ProvisioningSchemeAlreadyExists
The task failed because a provisioning scheme with the same name already exists.

IdentityPoolNotFound
The task failed because the specified identity pool could not be found.

MasterVMImagesNotPartOfProvisioningSchemeHostingUnit
The task failed because the hosting unit from which the master image originated is not the same hosting unit that the provisioning scheme is using.

MasterVMImagesAreSnapshots
The task failed because the master VM path does not refer to a Snapshot item.

ProvisioningSchemeNotDefined
The task failed because it could not find a provisioning scheme with the specified name.

TaskAlreadyRunningForProvisioningScheme
The task failed because a task for a provisioning scheme with the same name is already running.

InvalidMasterVMConfiguration
The task failed because the VM snapshot specified as the master has an invalid configuration.

InvalidMasterVMState
The task failed because the VM snapshot specified as the master is currently in an invalid state.

InsufficientResources
The task failed because the hypervisor did not have enough resources to complete the task.

DiskConsolidationFailed
The disk consolidation task failed. Details are in the task state information string.

StorageNotFound
The task failed because no associated storage was found in the hosting unit.

ConfigurationError
The task failed because the service is unable to contact one of the other services. This is because not all appropriate Configuration Service registrations have been performed.

Canceled
The task was stopped by user intervention (using Stop-ProvTask).

TaskStateInformation
Additional information about the current task state.

TaskProgress
The progress of the task 0-100%.

DiskSize
The size of the master image in GB
Notes
Only one long-running task for each provisioning scheme can be processed at a time.
In case of failure, the following errors can result.

Error Codes
-----------
JobCreationFailed
The requested task could not be started.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation. Communication with the database failed for various reasons.

MachineCreationServiceDoesNotSupportPersonalDisk
The service instance being used has not been upgraded to support the personal vDisk feature.

DatabaseMissingCapabilities
The database supporting the service instance being used has not been upgraded to support the personal vDisk feature.

CommunicationError
An error occurred while communicating with the service.

InvalidParameterCombination
Both PurgeJobOnSuccess and RunAsynchronously were specified. When running asynchronously, the cmdlet terminates before the job does, so it cannot clean up the completed job.

PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

ScopeNotFound
One or more of the scopes nominated for the new provisioning scheme do not exist.

WorkflowHostUnavailable
The task could not be started because the database connection is inactive.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

The cmdlet is associated with a task of type NewProvisioningScheme, and while active will move through the following operations (CurrentOperation field):

ValidatingInputs
ConsolidatingMasterImage
PreparingMasterImage
ReplicatingMasterImage
CommittingScheme

Examples
-------------------------- EXAMPLE 1 --------------------------
C:\PS> New-ProvScheme -ProvisioningSchemeName XenPS -HostingUnitName XenHu -IdentityPool
iName idPool1 -CleanOnBoot -MasterImageVM XDHyp:\HostingUnits\XenHU\Base.vm\Base.snapshot
TaskId                              : 90e93b9d-a225-4701-ad50-fa1546af35ac
Active                              : False
Host                                : MyHost
DateStarted                         : 17/05/2010 08:22:22
Type                                : NewProvisioningScheme
Metadata                            : {}
WorkflowStatus : Completed
ProvisioningSchemeName : XenPS
MasterImage : Base.vm/Base.snapshot
IdentityPoolName : idPool1
IdentityPoolUid : 03743136-e43b-4a87-a7f4-ab71668b3c16
HostingUnitName : XenHU
HostingUnitUid : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
PersonalVDiskDriveLetter : 
PersonalVDiskDriveSize : 0
ProvisioningSchemeUid : 7585d0de-192e-4847-a6d8-22713c3a2f42
CurrentOperation : 
TaskState : Finished
TaskStateInformation : 
TaskProgress : 100
DiskSize : 24

Creates a new provisioning scheme with the name “XenPS” using the hosting unit “XenHU” and the identity pool “idPool1” from the master VM snapshot called “Base.snapshot”.

-------------------------- EXAMPLE 2 --------------------------
C:\PS> New-ProvScheme -ProvisioningSchemeName XenPS -HostingUnitName XenHu -IdentityPoolName idPool1 -CleanOnBoot -MasterImageVM XDHyp\HostingUnits\XenHU\Base.vm\Base.snapshot -RunAsynchronously

Guid
----
6dd85f6e-96cf-46b1-9cd4-d8ba7d06e85b

Creates a new provisioning scheme with the name “XenPS” using the hosting unit “XenHU” and the identity pool “idPool1” from the master VM snapshot called “Base.snapshot” asynchronously. To get the task details, use Get-ProvTask -TaskID <task id>

i.e.
C:\PS>Get-ProvTask -TaskID 6dd85f6e-96cf-46b1-9cd4-d8ba7d06e85b
Task Id : 6dd85f6e-96cf-46b1-9cd4-d8ba7d06e85b
Active : False
Host : MyHost
Date Started : 17/05/2010 08:22:22
Type : NewProvisioningScheme
Metadata : {}
WorkflowStatus : Completed
ProvisioningSchemeName : XenPS
MasterImage : XDHyp\HostingUnits\XenHU\Base.vm\Base.snapshot
IdentityPoolName : idPool1
IdentityPoolUid : 03743136-e43b-4a87-a7f4-ab71668b3c16
HostingUnitName : XenHU
HostingUnitUid : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
PersonalVDiskDriveLetter : 
PersonalVDiskDriveSize : 0
ProvisioningSchemeUid : 7585d0de-192e-4847-a6d8-22713c3a2f42
Task State : Finished
Task State Information : 
Task Progress : 100
Disk Size : 24

-------------------------- EXAMPLE 3 --------------------------
C:\PS>$provScheme = New-ProvScheme -ProvisioningSchemeName XenPS2 -HostingUnitName XenHu -IdentityPoolName idPool1 -CleanOnBoot -MasterImageVM XDHyp\HostingUnits\XenHU\Base.vm\Base.snapshot -UsePersonalVDiskStorage -PersonalVDiskDriveSize 17 -PersonalVDiskDriveLetter x

Creates a new provisioning scheme with the name “XenPS2” using the hosting unit “XenHU” and the identity pool “idPool1” from the master VM snapshot called “Base.snapshot” synchronously. The personal vDisk is mapped as drive X. The operation runs synchronously, and the return value contains the task details

For example:
C:\PS>$provScheme

Task Id : d726222a-04b5-4098-b9ac-db85ed9351b
Active : False
Host : MyHost
DateStarted: 12/09/2011 09:30:04
Type: NewProvisioningScheme
Metadata: {}
ProvisioningSchemeName: XenPS2
IdentityPoolName: idPool1
IdentityPoolId: 03743136-e43b-4a87-af74-ab7686b3c16
HostingUnitName: XenHU
HostingUnitId: 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
PersonalVDiskDriveLetter: X
PersonalVDiskDriveSize: 17
WorkflowStatus: Completed
MasterImage: XDhyp\HostingUnitst\XenHU\Base.vm\Base.snapshot
ProvisioningSchemeId: 7585f0de-192e-4847-a6d8-22713c3a2f42
TaskState: Finished
TaskStateInformation:
TaskProgress: 100
DiskSize: 24
New-ProvVM

Oct 10, 2013
Creates a new virtual machine.

Syntax


Detailed Description

Lets you create new virtual machines with the configuration specified by a provisioning scheme.

The virtual machines are created using all of the storage specified in the provisioning scheme. The storage is used in a round robin mechanism. For the duration of this task, the AD accounts are locked (by the AD Identity Service), which stops the same accounts being used by other virtual machine creation tasks.

Related topics

Get-ProvVM
Remove-ProvVM
Lock-ProvVM
Unlock-ProvVM

Parameters

-ProvisioningSchemeName <String>
The name of the provisioning scheme in which the virtual machines are created.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-ProvisioningSchemeUid <Guid>
The unique identifier for the provisioning scheme in which the virtual machines are created.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |
-**ADAccountName**<String[]>

A list of the Active Directory account names that are used for the virtual machines. The accounts must be provided in a domain-qualified format. This parameter accepts Identity objects as returned by the New-AcctADAccount cmdlet, or any PSObject with string properties "Domain" and "ADAccountName".

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**FastBuild**<SwitchParameter>

Indicates whether or not the command can improve speed by using optimizations in the creation process. This may mean that machines you create cannot be booted until ResetVM operations have been performed by the Broker and Machine Identity Services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**NetworkMapping**<Hashtable>

Specifies how the attached NICs are mapped to networks. If this parameter is omitted, provisioned VMs are created with network settings as inherited from the provisioning scheme. If this parameter is supplied, machines are created with the number of NICs specified in the map, and each NIC is attached to the specified network.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**RunAsynchronously**<SwitchParameter>

Indicates whether or not the command returns before it is complete. If specified, the command returns an identifier for the task that was created. This task can be monitored using the get-ProvTask command.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-**PurgeJobOnSuccess**<SwitchParameter>
Indicates that the task history is removed from the database when the task has finished. This cannot be specified for tasks that are run asynchronously.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId**:Guid

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**:String

Specifies the address of a XenDesktop controller to which the PowerShell snap-in connects. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.Guid

When the RunAsynchronously identifier is specified, this GUID is returned and provides the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task that was run and contains the following information:

TaskId :Guid

The identifier for the task that was performed.

Active :Boolean

Indicates whether the task is still processing or is complete.

Host :string
The name of the host on which the task is running or was run.

DateStarted <DateTime>

The date and time that the task was initiated.

Type <Citrix.XDInterServiceTypes.JobType>

The type of task. For newly provisioned VM tasks, this is always "NewVirtualMachine".


The list of metadata stored against the task. For new tasks this is empty until metadata is added.


Indicates the status of the workflow that is used to process the task.

ProvisioningSchemeName <string>

The name of the provisioning scheme that the task was for.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme that the task was for.

MasterImage <string>

The path (in the hosting unit provider) of the virtual machine snapshot that was used as the master image for the task.

IdentityPoolName <string>

The name of the identity pool (from the ADIdentity PowerShell snap-in) that the new provisioning scheme uses.

IdentityPoolUid <guid>

The unique identifier name of the identity pool (from the ADIdentity PowerShell snap-in) that the new provisioning scheme uses.

HostingUnitName <string>

The name of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme uses.

HostingUnitUid

The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme uses.

TaskState <Citrix.DesktopUpdateManager.SDK.ProvisionVMState>

The state of the task. This can be any of the following:

Processing

Indicates that the task is in its initial state.

LocatingResources,
Indicates that the task is locating information from other services.

HostingUnitNotFound

Indicates that the required hosting unit could not be located.

ProvisioningSchemeNotFound

Indicates that the required provisioning scheme could not be located.

Provisioning

Indicates that the task is at the provisioning stage.

IdentityPoolNotFound

Indicates that the required identity pool could not be located.

TaskAlreadyRunningForProvisioningScheme

Indicates that the provisioning scheme already has another task running.

HypervisorInMaintenanceMode

Indicates that the hypervisor is not available for normal use.

NoAvailableStorage

Indicates that no storage is available for the hypervisor.

NoAvailableNetwork

Indicates that no network is available for the hypervisor.

Finalizing

Indicates that the task is finalizing.

Finished

Indicates that the task is complete.

FinishedWithErrors

Indicates that the task is complete but there were errors. Specific details of errors are included with each failed virtual machine.

Removing

Indicates that the task is removing virtual machines from the hypervisor.

Failed

The job failed for reasons specified in TaskStateInformation.

Canceled
Indicates that the task was stopped by user intervention (using Stop-ProvTask).

TaskStateInformation

Provides more detailed information about the current task state.

VirtualMachinesToCreateCount <int>

The total number of virtual machines that the task is trying to create.

VirtualMachinesCreatedCount <int>

The number of virtual machines that the task has created so far. Details of the machines that were created are in the CreatedVirtualMachines parameter.

VirtualMachinesCreationFailedCount <int>

The number of virtual machines that the task has failed to create. Details of the machines that were not created are in the FailedVirtualMachines parameter.

FailedVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>

ADAccountName <string>

The domain qualified AD Account name of the machine.

ADAccountSid <string>

The AD account SID of the machine account.

DiagnosticInformation <Citrix.MachineCreation.Sdk.ExceptionSurrogate[]>

A collection of handled error states which caused the provisioning to fail.

ExceptionType <string>

The type of exception this object represents.

Message <string>

The exception message.

Details <string>

The full exception content including stack trace.

InnerException <Citrix.MachineCreation.Sdk.ExceptionSurrogate>

Information relating to any contributing error state.

Status <string>

StatusAdditionalInformation <string>

VMId <string>
The virtual machine identifier within the hypervisor in which the VM resides.

VMName <string>
The display name of the virtual machine within the hypervisor in which the VM resides.

CreatedVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>
See FailedVirtualMachines for details of the object parameters.

Notes
Only one long-running task in each provisioning scheme can be processed at a time.
In the case of failure, the following errors can result.

Error Codes
-----------

JobCreationFailed
The requested task could not be started.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation. Communication with the database failed for various reasons.

WorkflowHostUnavailable
The task could not be started because the database connection is inactive.

PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error

CommunicationError
An error occurred while communicating with the service.

InvalidParameterCombination
Both PurgeJobOnSuccess and RunAsynchronously were specified.
When running asynchronously, the cmdlet terminates before the job does,
so it cannot clean up the completed job.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs
on the controller being used, or examine the XenDesktop logs.

The cmdlet is associated with a task of type NewVirtualMachine, and while active will move through the following
operations (CurrentOperation field)

ValidatingInputs
CreatingVirtualMachines
ReleasingAccountLocks

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> New-provVM -ProvisioningSchemeName MyScheme -ADAccountName "domain\Account"

TaskId                             : 4b49f13a-277c-4cb0-bc40-f088430cfe8a
Active                             : False
Host                               : DUMTESTDDC
DateStarted                        : 18/05/2010 10:54:32
Type                               : NewVirtualMachine
Metadata                           : {}
WorkflowStatus                     : Completed
MasterImage                        : /Base.vm/Base.snapshot
ProvisioningSchemeName             : MyScheme
ProvisioningSchemeUid             : 7585f0de-192e-4847-a6d8-22713c3a2f42
CurrentOperation                   :
TaskState                          : Finished
TaskStateInformation               :
HostingUnitUid                     : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
HostingUnitName                    : XenHU
IdentityPoolUid                    : 03743136-e43b-4a87-af74-ab71686b3c16
IdentityPoolName                   : idPool1
VirtualMachinesToCreateCount       : 1
VirtualMachinesCreatedCount        : 1
VirtualMachinesCreationFailedCount : 0
CreatedVirtualMachines             : {DOMAIN\Account$}
FailedVirtualMachines : {}
ProvisioningJob : 6fa5dc7c-6d49-4616-8682-aeb1580866b3
ProvisioningStatus : Completed

Creates a new virtual machine using the AD account "domain\account" in the provisioning scheme "MyScheme".

-------------------------- EXAMPLE 2 --------------------------

C:\PS>New-provVM -ProvisioningSchemeName MyScheme -ADAccountName "domain\Account" -RunAsynchronously

Guid ----
6dd85fec-96cf-46b1-9cd4-d8ba7d06e85b

Creates a new virtual machine using the AD account "domain\account" in the provisioning scheme "MyScheme" asynchronously. To get the task details, use Get-ProvTask -TaskID <task id>

i.e.

C:\PS>Get-ProvTask -TaskID 6dd85fec-96cf-46b1-9cd4-d8ba7d06e85b

TaskId : 4b49f13a-277c-4cb0-bc40-f088430cfe8a
Active : False
Host : MyHost
DateStarted : 18/05/2010 10:54:32
Type : NewVirtualMachine
Metadata : {}
WorkflowStatus : Completed
MasterImage : /Base.vm/Base.snapshot
ProvisioningSchemeName : MyScheme
ProvisioningSchemeUid : 7585f0de-192e-4847-a6d8-22713c3a2f42
TaskState : Finished
TaskStateInformation :
HostingUnitUid : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
HostingUnitName : XenHU
IdentityPoolUid : 03743136-e43b-4a87-af74-ab71686b3c16
IdentityPoolName : idPool1
VirtualMachinesToCreateCount : 1
VirtualMachinesCreatedCount : 1
VirtualMachinesCreationFailedCount : 0
CreatedVirtualMachines : {DOMAIN\Account$}
FailedVirtualMachines : {}
ProvisioningJob : 6fa5dc7c-6d49-4616-8682-aeb1580866b3
ProvisioningStatus : Completed

-------------------------- EXAMPLE 3 --------------------------

C:\PS>$accounts = Get-AcctAdAccount -IdentityPool MyPool -State Available
C:\PS>New-provVM -ProvisioningSchemeName MyScheme -ADAccountName $t -RunAsyncronously

Creates a new virtual machine using all of the available AD accounts from the identity pool "MyPool" in the provisioning scheme "MyScheme" asynchronously. To get the task details, use Get-ProvTask -TaskID <task id>

For example:

C:\PS>Get-ProvTask -TaskID 6dd85fec-96cf-46b1-9cd4-d8ba7d06e85b
TaskId : 4b49f13a-277c-4cb0-bc40-f088430cfe8a
Active : False
Host : MyHost
DateStarted : 18/05/2010 10:54:32
Type : NewVirtualMachine
Metadata : {}
WorkflowStatus : Completed
MasterImage : /Base.vm/Base.snapshot
ProvisioningSchemeName : MyScheme
ProvisioningSchemeUid : 7585f0de-192e-4847-a6d8-22713c3a2f42
TaskState : Finished
TaskStateInformation :
HostingUnitUid : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
HostingUnitName : XenHU
IdentityPoolUid : 03743136-e43b-4a87-af74-ab71686b3c16
IdentityPoolName : idPool1
VirtualMachinesToCreateCount : 1
VirtualMachinesCreatedCount : 1
VirtualMachinesCreationFailedCount : 0

CreatedVirtualMachines : {DOMAIN\Account$}

FailedVirtualMachines : {} 

ProvisioningJob : 6fa5dc7c-6d49-4616-8682-aeb1580866b3

ProvisioningStatus : Completed
Publish-ProvMasterVmImage

Oct 10, 2013
Update the master image associated with the provisioning scheme.

Syntax

Publish-ProvMasterVmImage [-ProvisioningSchemeName] <String> -MasterImageVM <String> [-DoNotStoreOldImage] [-RunAsynchronously] [-PurgeJobOnSuccess] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Publish-ProvMasterVmImage -ProvisioningSchemeUid <Guid> -MasterImageVM <String> [-DoNotStoreOldImage] [-RunAsynchronously] [-PurgeJobOnSuccess] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to update the hard disk image used to provision virtual machines. If the provisioning scheme is a 'CleanOnBoot' type, then the next time that virtual machines are started, their hard disks are updated to this new image. Regardless of the 'CleanOnBoot' type, all new virtual machines created after this command will use this new hard disk image.

A background task will be created to remove the old hard disk copies. You can locate and monitor this task using the Get-ProvTask cmdlet.

A snapshot is used rather than a VM, so that the content of the hard disk for the provisioning scheme can be easily determined.

As the snapshot is specified using a path into the Citrix Host Service Powershell Provider, the Citrix Host Service Powershell snap-in must also be loaded for this cmdlet to be used.

The previous hard disk image path is stored into the history (see Get-provSchemeMasterVMImageHistory). The data stored in the history allows for a roll back to be undertaken, to revert to the previous hard disk image if required.

Related topics

Get-ProvSchemeMasterVMImageHistory

Get-ProvScheme

Parameters

-ProvisioningSchemeName <String>

The provisioning scheme to which the hard disk image should be updated.

| Required?  | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-ProvisioningSchemeUid <Guid>
The provisioning scheme identifier to which the hard disk image should be updated.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

**-MasterImageVM:String**

The path in the hosting unit provider to the virtual machine snapshot that will be used. This identifies the hard disk to be used and the default values for the memory and processors. This must be a path to a Snapshot Item in the same hosting unit that the hosting unit name or hosting unit Uid refers to.

Valid paths are of the format: XDHyp:\HostingUnits\<HostingUnitName>\<path>\<VmName>.vm\<SnapshotName>.snapshot

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

**-DoNotStoreOldImage:SwitchParameter**

Specifies that the current image should not be added to the image history list for the provisioning scheme. This is useful when rolling back to a previous image.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-RunAsynchronously:SwitchParameter**

Indicates whether or not the cmdlet should return before it is complete. If specified, the command returns an identifier for the task that was created. You can monitor this task using the get-ProvTask cmdlet.

| Required? | false |
| Default Value | false |
| Accept Pipeline Input? | false |

**-PurgeJobOnSuccess:SwitchParameter**

Indicates that the task history will be removed from the database once the task has finished. This cannot be specified for tasks that are run asynchronously.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId <Guid>**

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress <String>**

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.Guid

When "RunAsynchronously" is specified, this identifier is returned to provide the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task run and contains the following information:

TaskId <Guid>

The identifier for the task that was performed.

Active <Boolean>

Indicates whether the task is still processing or is complete.

Host <string>

The name of the host on which the task is running or was run.

DateTime <DateTime>
The date and time that the task was initiated.

Type <Citrix.XDInterServiceTypes.JobType>
The type of the task. For new provisioning scheme tasks this is always "NewProvisioningScheme".

The list of metadata stored against the task. For new tasks this will be empty until metadata is added.

Indicates the status of the workflow that is being used to process the task.

ProvisioningSchemeName <string>
The name of the provisioning scheme that the task was for.

ProvisioningSchemeUid <Guid>
The unique identifier of the provisioning scheme that the task was for.

MasterImage <string>
The path (in the hosting unit provider) of the virtual machine snapshot that was used as the master image for the task.

IdentityPoolName <string>
The name of the identity pool (from the ADIdentity PowerShell snap-in) that the new provisioning scheme will use.

IdentityPoolUid <guid>
The unique identifier name of the identity pool (from the ADIdentity PowerShell snap-in) that the new provisioning scheme will use.

HostingUnitName <string>
The name of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme will use.

HostingUnitUid
The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme will use.

TaskState <Citrix.DesktopUpdateManager.SDK.NewProvisioningSchemeState>
The state of the task. This can be any of the following:

Processing
Indicates that the task has just begun and has not done anything yet.

LocatingResources,
Indicates that the workflow is locating resources from other services.
HostingUnitNotFound
Indicates that the task failed because the required hosting unit could not be located.

VirtualMachineSnapshotNotFound
Indicates that the task failed because the required snapshot could not be located.

ConsolidatingMasterImage
Indicates that the task is consolidating the master image.

ReplicatingConsolidatedImageToAllStorage
Indicates that the task is replicating the consolidated master image.

StoringProvisioningScheme
Indicates that the task is storing the provisioning scheme data to the database.

Finished
Indicates that the task has completed with no errors.

ProvisioningSchemeAlreadyExists
Indicates that the task failed because a provisioning scheme with the same name already exists.

IdentityPoolNotFound
Indicates that the task failed because the identity pool specified could not be found.

MasterVMImageIsNotPartOfProvisioningSchemeHostingUnit,
Indicates that the hosting unit that the master image originated from is not the same hosting unit that the provisioning scheme is defined to use.

MasterVMImageIsNotASnapshot
Indicates that the task failed because the master VM Image path does not refer to a Snapshot item.

ProvisioningSchemeNotFound
Could not find a provisioning scheme with the specified name.

TaskAlreadyRunningForProvisioningScheme
A task for a provisioning scheme with the same name is already running.

InvalidMasterVMConfiguration
The VM snapshot specified as the master had an invalid configuration.

InvalidMasterVMState
The VM snapshot specified as the master is currently in an invalid state.
InsufficientResources
Indicates that the task failed because the hypervisor did not have enough resources to complete the task.

StorageNotFound
Indicates that no associated storage was found in the hosting unit.

Canceled
Indicates that the task was stopped by user intervention (using Stop-ProvTask)

TaskStateInformation <string>
Holds string data providing more details about the current task state.

TaskProgress
The progress of the task 0-100%.

Notes
Only one long running task per provisioning scheme can be processed at any one time.

In the case of failure, the following errors can result.

Error Codes
---------

JobCreationFailed
The requested task could not be started.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for
for various reasons.

WorkflowHostUnavailable
The task could not be started because the database connection is inactive.

CommunicationError
An error occurred while communicating with the service.
PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationExceptionLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

The cmdlet is associated with a task of type PublishImage, and while active will move through the following operations (CurrentOperation field)

ValidatingInputs
ConsolidatingMasterImage
PreparingMasterImage
ReplicatingMasterImage
CommittingScheme

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Publish-ProvMasterVMImage -ProvisioningSchemeName MyScheme -MasterImageVM XDHyp:\HostUnits\HostUnit1\RhoneCC_baseXP.vm\base.snapshot

TaskId : 248f102f-073f-45fe-aea9-1819a6d6dd1f
Active : False
Host : MyHost
DateStarted : 17/05/2010 17:37:57
Type : PublishImage
Metadata : {}
WorkflowStatus : Completed
ProvisioningSchemeUid : 7585f0de-192e-4847-a6d8-22713c3a2f42
ProvisioningSchemeName : MyScheme
MasterImage : /RhoneCC_baseXP.vm\base.snapshot
HostingUnitName : HostUnit1
HostingUnitUid : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
ADIdentityPoolName : 
ADIdentityPoolUid : 03743136-e43b-4a87-af74-ab71686b3c16
CurrentOperation :
TaskState : Finished
TaskStateInformation :

Updates the master hard disk image for the provisioning Scheme "MyScheme" to use the "base.snapshot" hard disk image.
Remove-ProvScheme

Oct 10, 2013
Removes a provisioning scheme

Syntax

Remove-ProvScheme [-ProvisioningSchemeName] <String> [-ForgetVM] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-ProvScheme -ProvisioningSchemeUid <Guid> [-ForgetVM] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove a provisioning Scheme. The provisioning scheme must not contain any VMs unless the ‘ForgetVM’ option is specified.

If ‘ForgetVM’ is not specified, a cmdlet task is created that runs in the background to remove the hard disk copies that have been created for the provisioning scheme in hypervisor storage. Use the Get-ProvTask command to monitor the progress of this task.

Related topics

Get-ProvTask

New-ProvScheme

Parameters

-ProvisioningSchemeName <String>

The name of the provisioning scheme to be removed.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByPropertyName) |

-ProvisioningSchemeUid <Guid>

The unique identifier for the provisioning scheme to be removed.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByPropertyName) |

-ForgetVM <SwitchParameter>
Indicates whether or not the VMs in the provisioning scheme should be left in the hypervisor and only the data held in the Machine Creation Services removed. If this is specified, it is up to the administrator of the hypervisor to remove the VMs and hard disk images using the tools provided by the hypervisor itself.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.MachineCreation.Sdk.ProvisioningScheme

This object provides details of the provisioning scheme and contains the following information:

ProvisioningSchemeUid
The unique identifier for the provisioning scheme.

ProvisioningSchemeName
The name of the provisioning scheme.

CpuCount
The number of processors that VMs will be created with when using this scheme.
MemoryMB
The maximum amount of memory that VMs will be created with when using this scheme.

MasterImageVM
The path within the hosting unit provider to the VM or snapshot of which the scheme is currently using a copy.

MasterImageVMDate
The date and time that the copy of the VM image was made for the scheme.

IdentityPoolUid
The unique identifier of the identity pool (from the ADIdentity PowerShell snap-in) that the scheme uses.

IdentityPoolName
The name of the identity pool (from the ADIdentity PowerShell snap-in) that the scheme uses.

HostingUnitUid
The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme will use.

HostingUnitName
The name of the hosting unit (from the hosting unit PowerShell snap-in) that the new provisioning scheme will use.

CleanOnBoot
Indicates whether the VMs created are to be reset to a clean state on each boot.

TaskId
The identifier of any current task that is running for the provisioning scheme.

Notes
If the hosting unit referenced by the provisioning scheme no longer exists (i.e. it has been removed using the Hosting Unit PowerShell snap-in), the provisioning scheme data is deleted from the database without errors. However, the hard disks associated with the provisioning scheme cannot be removed and remain in the hypervisor.

In the case of failure, the following errors can result.

Error Codes
-------------

IllegalParameter
One or more parameters are illegal or are not specified.

ProvisioningSchemeNotFound
The specified provisioning scheme could not be located.
UnableToRemoveProvisioningSchemeDueToAssociatedVM
The provisioning scheme contained VMs and the 'ForgetVM' parameter was not specified.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

The cmdlet is associated with a task of type DisusedImageCleanUp, and while active will move through the following operations (CurrentOperation field)

Running

Examples

-------------------------- EXAMPLE 1 --------------------------
c:PS>Remove-ProvScheme -ProvisioningSchemeName $provScheme.ProvisioningSchemeName
Remove the empty provisioning scheme by name.
Remove-ProvSchemeControllerAddress

Oct 10, 2013
Removes metadata from a provisioning scheme.

Syntax
Remove-ProvSchemeControllerAddress [-ProvisioningSchemeName] <String> [-ControllerAddress] <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters]
Remove-ProvSchemeControllerAddress -ProvisioningSchemeUid <Guid> -ControllerAddress <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [-CommonParameters]

Detailed Description
Removes the specified controller addresses from the specified object. Attempting to remove an address not present writes an error record to the pipeline.

Related topics
Get-ProvScheme
Add-ProvSchemeControllerAddress

Parameters
-ProvisioningSchemeName <String>
The name of the provisioning scheme from which metadata is to be removed.

Required?  true
Default Value
Accept Pipeline Input?  true (ByPropertyName)

-ProvisioningSchemeUid <Guid>
The identifier of the provisioning scheme from which metadata is to be removed.

Required?  true
Default Value
Accept Pipeline Input?  false

-ControllerAddress <String[]>
Specifies the array of DNS names to be removed from the provisioning scheme.

Required?  true
Default Value
Accept Pipeline Input?  true (ByPropertyName)

-LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Required?  false
Default Value
Accept Pipeline Input?  false

-AdminAddress <String>
Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.
Required? | False
--- | ---
Default Value | LocalHost. Once a value is provided by any cmdlet, this value becomes the default.
Accept Pipeline Input? | False

**Input Type**

Citrix.MachineCreation.Sdk.ProvisioningScheme You can pipe an object containing a parameter called 'ProvisioningSchemeName' to Remove-ProvSchemeMetadata.

**Notes**

In the case of failure, the following errors can result.

**Error Codes**

-------------
ProvisioningSchemeNotFound
The specified provisioning scheme could not be located.

ControllerAddressNotFound
The specified address was not associated with the provisioning scheme.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

**Examples**

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Get-ProvScheme -ProvisioningSchemeName scheme1 | Remove-ProvSchemeControllerAddress

ProvisioningSchemeUid : 7585f0de-192e-4847-a6d8-22713c3a2f42
ProvisioningSchemeName : Scheme1
CpuCount               : 1
MemoryMB               : 1024
MasterImageVM          : Base.vm/Base.snapshot
MasterImageVMDate      : 17/05/2010 09:53:40
IdentityPoolUid        : 03743136-e43b-4a87-af74-ab71686b3c16
IdentityPoolName       : idPool1
HostingUnitUid         : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
HostingUnitName        : HostUnit1
CleanOnBoot            : True
TaskId                 : 00000000-0000-0000-0000-000000000000
Metadata               : {}
Remove all controller addresses from the provisioning scheme with the name "scheme1".

-------------------------- EXAMPLE 2 --------------------------

C:\PS> Remove-ProvSchemeControllerAddress -ProvisioningSchemeUid "01a4a008-8ce8-4165-ba9c-cdf15a6b0501" -ControllerAddress (ddcA.citrix.com,ddcC.citrix2.com)

ProvisioningSchemeUid : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
ProvisioningSchemeName : Scheme2
CpuCount : 1
MemoryMB : 1024
MasterImageVM : Base.vm/Base.snapshot
MasterImageVMDate : 17/05/2010 09:53:40
IdentityPoolUid : 03743136-e43b-4a87-af74-ab71686b3c16
IdentityPoolName : idPool1
HostingUnitUid : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
HostingUnitName : HostUnit1
CleanOnBoot : True
TaskId : 00000000-0000-0000-0000-000000000000
Metadata : {}
ControllerAddress : {ddcB.citrix.com}

Remove a subset of the controller address list from the provisioning scheme with the identifier "01a4a008-8ce8-4165-ba9c-cdf15a6b0501".
Remove-ProvSchemeMasterVMImageHistory

Oct 10, 2013
Removes the history of provisioning scheme master image VMs.

Syntax

Remove-ProvSchemeMasterVMImageHistory [-ProvisioningSchemeName] <String> [-VMImageHistoryUid <Guid>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-ProvSchemeMasterVMImageHistory -ProvisioningSchemeUid <Guid> [-VMImageHistoryUid <Guid>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-ProvSchemeMasterVMImageHistory -VMImageHistoryUid <Guid> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove the record of previously used master VMs or snapshots for provisioning schemes.

Related topics

Get-ProvSchemeMasterVMImageHistory

Publish-ProvMasterVMImage

Parameters

-ProvisioningSchemeName <String>
The name of the provisioning scheme.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-VMImageHistoryUid <Guid>

The unique identifier for the Image History item.
- **LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

**Notes**

In the case of failure, the following errors can result.

**Error Codes**

----------

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationException

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Remove-ProvSchemeMasterVMImageHistory -ProvisioningSchemeName MyScheme
Removes all history for the provisioning scheme called "MyScheme".

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-ProvScheme | Remove-ProvSchemeMasterVMImageHistory
Removes all history for all provisioning schemes.
Remove-ProvSchemeMetadata

Oct 10, 2013
Removes metadata from the given ProvisioningScheme.

Syntax

Remove-ProvSchemeMetadata [-ProvisioningSchemeUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-ProvSchemeMetadata [-ProvisioningSchemeUid] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-ProvSchemeMetadata [-ProvisioningSchemeName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-ProvSchemeMetadata [-ProvisioningSchemeName] <String> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-ProvSchemeMetadata [-InputObject] <ProvisioningScheme[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-ProvSchemeMetadata [-InputObject] <ProvisioningScheme[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given ProvisioningScheme.

Related topics

Add-ProvSchemeMetadata
Set-ProvSchemeMetadata

Parameters

-ProvisioningSchemeUid <Guid>
Id of the ProvisioningScheme

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-ProvisioningSchemeName <String>
Name of the ProvisioningScheme

<p>| Required? | true |</p>
<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true (ByValue, ByPropertyName)</td>
<td></td>
</tr>
</tbody>
</table>

**-InputObject**<ProvisioningScheme[]>

Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-Map**<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-Name**<String>

The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>
Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes
If the command fails, the following errors can be returned.

Error Codes
-----------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.
ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ProvScheme | Remove-ProvSchemeMetadata
Remove all metadata from all ProvisioningScheme objects.
Remove-ProvSchemeScope

Oct 10, 2013
Remove the specified ProvisioningScheme(s) from the given scope(s).

Syntax

Remove-ProvSchemeScope [-Scope] <String[]> -InputObject <ProvisioningScheme[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-ProvSchemeScope [-Scope] <String[]> -ProvisioningSchemeUid <Guid[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Remove-ProvSchemeScope [-Scope] <String[]> -ProvisioningSchemeName <String[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

The RemoveProvSchemeScope cmdlet is used to remove one or more ProvisioningScheme objects from the given scope(s).

There are multiple parameter sets for this cmdlet, allowing you to identify the ProvisioningScheme objects in different ways:

- ProvisioningScheme objects can be piped in or specified by the InputObject parameter
- The ProvisioningSchemeUid parameter specifies objects by ProvisioningSchemeUid
- The ProvisioningSchemeName parameter specifies objects by ProvisioningSchemeName (supports wildcards)

To remove a ProvisioningScheme from a scope you need permission to change the scopes of the ProvisioningScheme.

If the ProvisioningScheme is not in a specified scope, that scope will be silently ignored.

Related topics

Add-ProvSchemeScope
Get-ProvScopedObject

Parameters

- **Scope**<String[]>

  Specifies the scopes to remove the objects from

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **InputObject**<ProvisioningScheme[]>

  Specifies the ProvisioningScheme objects to be removed.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ProvisioningSchemeUid</td>
<td>true</td>
<td>&lt;Guid&gt;[]</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
<tr>
<td>-ProvisioningSchemeName</td>
<td>true</td>
<td>&lt;String&gt;[]</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
<tr>
<td>-LoggingId</td>
<td>false</td>
<td>&lt;Guid&gt;</td>
<td>false</td>
</tr>
<tr>
<td>-AdminAddress</td>
<td>false</td>
<td>&lt;String&gt; Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Specifies the ProvisioningScheme objects to be removed by ProvisioningSchemeUid.

Specifies the ProvisioningScheme objects to be removed by ProvisioningSchemeName.

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Return Values
Notes
If the command fails, the following errors can be returned.

Error Codes
--------

UnknownObject
One of the specified objects was not found.

ScopeNotFound
One of the specified scopes was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Remove-ProvSchemeScope Finance -ProvisioningSchemeUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
Removes a single ProvisioningScheme from the 'Finance' scope.
-------------------------- EXAMPLE 2 --------------------------

c:\PS>Remove-ProvSchemeScope Finance,Marketing -ProvisioningSchemeUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
Removes a single ProvisioningScheme from multiple scopes.

-------------------------- EXAMPLE 3 --------------------------

c:\PS>Get-ProvScheme | Remove-ProvSchemeScope Finance
Removes all visible ProvisioningScheme objects from the 'Finance' scope.

-------------------------- EXAMPLE 4 --------------------------

c:\PS>Remove-ProvSchemeScope Finance -ProvisioningSchemeName A*
Removes ProvisioningScheme objects with a name starting with an 'A' from the 'Finance' scope.
Remove-ProvServiceConfigurationData

Oct 10, 2013
Removes configuration data from the service.

Syntax

Remove-ProvServiceConfigurationData [\[-Name\] <String>] [\-LoggingId <Guid>] [\-AdminAddress <String>] [\<CommonParameters>]

Detailed Description

Provides the ability to remove data from the MachineCreation Service configuration data.

Related topics

Set-ProvServiceConfigurationData
Get-ProvServiceConfigurationData

Parameters

-**Name**<String>

The name of the configuration data item to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default.
--- | ---
Accept Pipeline Input? | false

Notes
In the case of failure the following errors can be produced.

Error Codes
-----------

**DatabaseError**
An error occurred in the service while attempting a database operation.

**DatabaseNotConfigured**
The operation could not be completed because the database for the service is not configured.

**DataStoreException**
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

**PermissionDenied**
You do not have permission to execute this command.

**AuthorizationError**
There was a problem communicating with the Citrix Delegated Administration Service.

**ConfigurationLoggingError**
The operation could not be performed because of a configuration logging error.

**CommunicationError**
An error occurred while communicating with the service.

**ExceptionThrown**
An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------

C:\PS>Remove-ProvServiceConfigurationData -Name "MyName"
Removes the configuration data item with name "MyName".
Remove-ProvServiceMetadata

Oct 10, 2013
Removes metadata from the given Service.

Syntax

Remove-ProvServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-ProvServiceMetadata [-ServiceHostId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-ProvServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-ProvServiceMetadata [-InputObject] <Service[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given Service.

Related topics
Set-ProvServiceMetadata

Parameters

-**ServiceHostId**<Guid>

Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-**InputObject**<Service[]>

Objects to which metadata is to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Map**<PSObject>
Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name**: String

The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**: Guid

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**: String

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

Error Codes

---------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

---------------------------- EXAMPLE 1 ----------------------------

c:\PS>Get-ProvService | % { Remove-ProvServiceMetadata -Map $_.MetadataMap }
Remove all metadata from all Service objects.
Remove-ProvTask

Oct 10, 2013
Removes from the database completed tasks for the MachineCreation Service.

Syntax

Remove-ProvTask [-TaskId] <Guid> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Enables completed tasks that have run within the MachineCreation Service to be removed from the database.

Related topics

Get-ProvTask
Add-ProvTaskMetadata
Remove-ProvTaskMetadata

Parameters

-TaskId <Guid>

Specifies the identifier for the task to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
</table>
Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default.
--- | ---
Accept Pipeline Input? | false

**Input Type**

`System.Management.Automation.PSObject` Objects containing the TaskId parameter can be piped to the Remove-ProvTask command.

**Notes**

If the command fails, the following errors can result.

**Error Codes**

```
---------
InvalidParameterCombination
The cmdlet parameters are inconsistent.
InvalidWorkflow
The specified task could not be found.
InvalidWorkflowState
The task was not in an appropriate state for the requested operation.
DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.
PermissionDenied
You do not have permission to execute this command.
AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.
CommunicationError
```
There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-ProvTask -active $false | Remove-ProvTask
Success
Remove entries for all completed tasks from the MachineCreation Service.
Remove-ProvTaskMetadata

Oct 10, 2013
Removes metadata from the given Task.

Syntax

Remove-ProvTaskMetadata [-TaskId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-ProvTaskMetadata [-TaskId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-ProvTaskMetadata [-InputObject] <Task[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-ProvTaskMetadata [-InputObject] <Task[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given Task.

Related topics

Add-ProvTaskMetadata
Set-ProvTaskMetadata
Get-ProvTask
Remove-ProvTask
Stop-ProvTask
Switch-ProvTask

Parameters

-TaskId <Guid>
Id of the Task

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-InputObject <Task[]>
Objects to which the metadata is to be added.
<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th><strong>true</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><strong>true (ByValue)</strong></td>
</tr>
</tbody>
</table>

- **Map**<PSObject>

  Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with `@{"name1" = "val1"; "name2" = "val2"}`) or a string dictionary (created with `new-object System.Collections.Generic.Dictionary[String,String]`).

<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th><strong>true</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><strong>true (ByValue)</strong></td>
</tr>
</tbody>
</table>

- **Name**<String>

  The metadata property to remove.

<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th><strong>true</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><strong>false</strong></td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th><strong>false</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><strong>false</strong></td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th><strong>Required?</strong></th>
<th><strong>false</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default Value</strong></td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td><strong>Accept Pipeline Input?</strong></td>
<td><strong>false</strong></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | false

Notes

If the command fails, the following errors can be returned.

Error Codes

---------

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ProvTask | Remove-ProvTaskMetadata
Remove all metadata from all Task objects.
Remove-ProvVM

Oct 10, 2013
Removes virtual machines.

Syntax

Remove-ProvVM [-ProvisioningSchemeName] <String> -VMName <String[]> [-ForgetVM] [-RunAsynchronously] [-PurgeJobOnSuccess] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]


Detailed Description

Lets you remove VMs from the Machine Creation Services and the hypervisor that they run on.

Related topics
Get-ProvVM
New-ProvVM

Parameters

-ProvisioningSchemeName <String>

The name of the provisioning scheme from which virtual machines will be removed.

| Required? | true |
| Default Value |  |
| Accept Pipeline Input? | true (ByPropertyName) |

-ProvisioningSchemeUid <Guid>

The unique identifier for the provisioning scheme from which the virtual machines are removed.

| Required? | true |
| Default Value |  |
| Accept Pipeline Input? | true (ByPropertyName) |

-VMName <String[]>

List of VM names that will be removed from the provisioning scheme.

<p>| Required? | true |
| Default Value |  |</p>
<table>
<thead>
<tr>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

**-ForgetVM**<SwitchParameter>

The named VMs will only be removed from the provisioning scheme database, but will remain in the hypervisor.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-RunAsynchronously**<SwitchParameter>

Indicates whether or not the command returns before it is complete. If this is specified, the command returns an identifier for the task that was created. This task can be monitored using the get-ProvTask command.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-PurgeJobOnSuccess**<SwitchParameter>

Indicates that the task history is removed from the database when the task finishes. This cannot be specified for tasks that are run asynchronously.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>
Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. When a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.Guid

When the RunAsynchronously identifier is specified, this GUID is returned and provides the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task that was run and contains the following information:

TaskId <Guid>

The identifier for the task that was performed.

Active <Boolean>

Indicates whether the task is still processing or is complete.

Host <string>

The name of the host on which the task is running or was run.

DateStarted <DateTime>

The date and time that the task was initiated.

Type <Citrix.XDInterServiceTypes.JobType>

The type of task. For new remove-VM tasks, this is always "RemoveVirtualMachine".


The list of metadata stored for the task. For new tasks, this is empty until metadata is added.


Indicates the status of the workflow that is used to process the task.

ProvisioningSchemeName <string>

The name of the provisioning scheme that the task is for.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme that the task is for.
TaskState &lt;Citrix.DesktopUpdateManagerSDK.ProvisionVMState&gt;

The state of the task. This can be any of the following:

Processing
Indicates that the task is in its initial state.

LocatingResources,
Indicates that the task is locating information from other services.

HostingUnitNotFound
Indicates that the required hosting unit could not be located.

ProvisioningSchemeNotFound
Indicates that the required provisioning scheme could not be located.

Provisioning
Indicates that the task is at the provisioning stage.

IdentityPoolNotFound
Indicates that the required identity pool could not be located.

TaskAlreadyRunningForProvisioningScheme
Indicates that the provisioning scheme already has another task running.

Finalizing
Indicates that the task is finalizing.

Finished
Indicates that the task is complete.

FinishedWithErrors
Indicates that the task is complete but there were errors. Specific details of errors are included with each failed virtual machine.

Removing
Indicates that the task is removing virtual machines from the hypervisor.

Failed
Job failed for reasons specified in TaskStateInformation.

Canceled
Indicates that the task was stopped by user intervention (using Stop-ProvTask)
TaskStateInformation

Provides more detailed information about the current task state.

VirtualMachinesToRemoveCount <int>

The total number of virtual machines that the task is trying to remove.

VirtualMachinesRemovedCount <int>

The number of virtual machines that the task has removed so far. Details of the machines that have been removed are in the RemovedVirtualMachines parameter.

VirtualMachinesFailedCount <int>

The number of virtual machines that the task has failed to remove. Details of the machines that failed are in the FailedVirtualMachines parameter.

FailedVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>

ADAccountName <string>

The domain-qualified AD Account name of the machine.

ADAccountSid <string>

The AD account SID of the machine account.

DiagnosticInformation <Citrix.MachineCreation.Sdk.ExceptionSurrogate[]>

A collection of handled error states which caused the provisioning to fail.

ExceptionType <string>

The type of exception this object represents

Message <string>

The exception message

Details <string>

The full exception content including stack trace

InnerException <Citrix.MachineCreation.Sdk.ExceptionSurrogate>

Information relating to any contributing error state

Status <string>

StatusAdditionalInformation <string>

VMId <string>

The virtual machine identifier in the hypervisor.
VMName <string>
The display name of the virtual machine in the hypervisor.

RemovedVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>
See FailedVirtualMachines for details of the object parameters.

ProgressEstimator
Gives an estimate of the number of virtual machines processed, averaging over virtual machines that were both partly and completely processed.

Notes
Only one long-running task for each provisioning scheme can be processed at a time.

This task still operates if the hosting unit or VMs in the hypervisor are missing. This removes the data from the Citrix Machine Creation Services, but the VMs remain in the hypervisor.

In the case of failure, the following errors can result.

Error Codes
-----------
JobCreationFailed
The requested task could not be started.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation. Communication with the database failed for various reasons.

WorkflowHostUnavailable
The task could not be started because the database connection is inactive.

CommunicationError
An error occurred while communicating with the service.

InvalidParameterCombination
Both PurgeJobOnSuccess and RunAsynchronously were specified. When running asynchronously, the cmdlet terminates before the job does, so it cannot clean up the completed job.
PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationException

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

The cmdlet is associated with a task of type RemoveVirtualMachine, and while active will move through the following operations (CurrentOperation field)

ValidatingInputs

RemovingVirtualMachines

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>,(Get-ProvVM -ProvisioningSchemeName XenPS) | Remove-ProvVM XenPS

TaskId       : cfb506a5-cc7e-4a49-ac7b-dd960029d0d3
Active       : False
Host         : DDC
DateStarted  : 10/10/2012 16:39:45
Metadata     : {}
WorkflowStatus: Completed
ProvisioningSchemeUid : e1afc8fb-3f52-42ea-9a17-305fdb0b6ee4
ProvisioningSchemeName : XenPS
TaskState    : Finished
TaskStateInformation : 
VirtualMachinesToRemoveCount : 2
RemovedVirtualMachines : {XD\IP0001$, XD\IP0002$}
FailedVirtualMachines : {}
VirtualMachinesRemovedCount : 2
VirtualMachinesFailedCount : 0
ProgressEstimator : 2
Type          : RemoveVirtualMachine
Status        : Finished
CurrentOperation : 
TaskProgress  : 100
TaskExpectedCompletion : 10/10/2012 16:39:50
LastUpdateTime : 10/10/2012 16:39:50
ActiveElapsedTime : 5
DateFinished  : 10/10/2012 16:39:50
TerminatingError :
Remove all VMs from provisioning scheme XenPS
Rename-ProvScheme

Oct 10, 2013
Renames a provisioning scheme.

Syntax

Rename-ProvScheme [-ProvisioningSchemeName] <String> [-NewProvisioningSchemeName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Rename-ProvScheme -ProvisioningSchemeUid <Guid> [-NewProvisioningSchemeName] <String> [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to change the name of an existing provisioning scheme. The unique identifier for the provisioning scheme never changes after it has been created so, regardless of any name changes, the provisioning scheme can always be identified by its unique identifier.

Related topics

New-ProvScheme
Set-ProvScheme
Get-ProvScheme
Test-ProvSchemeNameAvailable

Parameters

-ProvisioningSchemeName <String>
The current name of the provisioning scheme.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-ProvisioningSchemeUid <Guid>
The identifier for the provisioning scheme that is to be renamed.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-NewProvisioningSchemeName <String>
The new name for the provisioning scheme. This must be a name that is not being used by an existing provisioning scheme, and it must not contain any of the following characters: `\/:#.*?=<>|[]"'``

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

**-PassThru**<SwitchParameter>

Defines whether or not the command returns a result showing the new state of the updated identity pool.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | true |

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | LocalHost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

**Return Values**

Citrix.MachineCreation.Sdk.ProvisioningScheme

This object provides details of the provisioning scheme and contains the following information:

ProvisioningSchemeUid

The unique identifier for the provisioning scheme.
ProvisioningSchemeName
The name of the provisioning scheme.

CpuCount
The number of processors that VMs will be created with when using this scheme.

MemoryMB
The maximum amount of memory that VMs will be created with when using this scheme.

MasterImageVM
The path within the hosting unit provider to the VM or snapshot of which the scheme is currently using a copy.

MasterImageVMDate
The date and time that the copy of the VM image was made for the scheme.

IdentityPoolUid
The unique identifier of the identity pool (from the ADIdentity PowerShell snap-in) that the scheme uses.

IdentityPoolName
The name of the identity pool (from the ADIdentity PowerShell snap-in) that the scheme uses.

HostingUnitUid
The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme will use.

HostingUnitName
The name of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme will use.

CleanOnBoot
Indicates whether the VMs created are to be reset to a clean state on each boot.

TaskId
The identifier of any current task that is running for the provisioning scheme.

Metadata
The metadata for the provisioning scheme.

Notes
In the case of failure, the following errors can result.

Error Codes
-----------

ProvisioningSchemeNotFound
The specified provisioning scheme could not be located.

ProvisioningSchemeNameAlreadyExists

A provisioning scheme with the same name as the new provisioning scheme name already exists.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. To locate more details, see the windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Rename-ProvScheme -provisioningSchemeName CurrentName -NewProvisioningSchemeName NewName
Renames a provisioning scheme from "currentName" to "NewName".

-------------------------- EXAMPLE 2 --------------------------

C:\PS>Rename-ProvScheme -provisioningSchemeName CurrentName -NewProvisioningSchemeName NewName -PassThru

ProvisioningSchemeUid : 7585f0de-192e-4847-a6d8-22713c3a2f42
ProvisioningSchemeName : NewName
CpuCount : 1
MemoryMB : 1024
MasterImageVM : /Base.vm
MasterImageVMDate : 17/05/2010 08:27:50
<table>
<thead>
<tr>
<th>IdentityPoolUid</th>
<th>03743136-e43b-4a87-af74-ab71686b3c16</th>
</tr>
</thead>
<tbody>
<tr>
<td>IdentityPoolName</td>
<td>IdPool1</td>
</tr>
<tr>
<td>HostingUnitUid</td>
<td>01a4a008-8ce8-4165-ba9c-cdf15a6b0501</td>
</tr>
<tr>
<td>HostingUnitName</td>
<td>XenHU</td>
</tr>
<tr>
<td>CleanOnBoot</td>
<td>True</td>
</tr>
<tr>
<td>TaskId</td>
<td></td>
</tr>
<tr>
<td>Metadata</td>
<td>{}</td>
</tr>
</tbody>
</table>

Renames a provisioning scheme from "currentName" to "NewName" and displays the resulting state.
Reset-ProvServiceGroupMembership

Oct 10, 2013
Reloads the access permissions and configuration service locations for the MachineCreation Service.

Syntax

Reset-ProvServiceGroupMembership [-ConfigServiceInstance] <ServiceInstance[]> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description
Enables you to reload MachineCreation Service access permissions and configuration service locations. The Reset-ProvServiceGroupMembership command must be run on at least one instance of the service type (Prov) after installation and registration with the configuration service. Without this operation, the MachineCreation services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-ProvServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Related topics

Parameters

- **-ConfigServiceInstance**<ServiceInstance[]>

  Specifies the configuration service instance object that represents the service instance for the type 'InterService' that references a configuration service for the deployment.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-LoggingId**<Guid>

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

- **-AdminAddress**<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Input Type
Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-ProvServiceGroupMembership command.

Notes
If the command fails, the following errors can be returned.

Error Codes
---------

NoSuitableServiceInstance
None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-ProvServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config -AdminAddress OtherServer.example.com | Reset-ProvServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named ‘OtherServer.example.com’.
Set-ProvDBConnection

Oct 10, 2013
Configures a database connection for the MachineCreation Service.

Syntax

Set-ProvDBConnection [-DBConnection] <String> [-Force] [-LoggingId <Guid>] [-AdminAddress <String>] [ [<CommonParameters>]

Detailed Description

Configures a connection to a database in which the MachineCreation Service can store its state. The service will attempt to connect and start using the database immediately after the connection is configured. The database connection string is updated to the specified value regardless of whether it is valid or not. Specifying an invalid connection string prevents a service from functioning until the error is corrected.

After a connection is configured, you cannot alter it without first clearing it (by setting the connection to $null).

You do not need to configure a database connection to use this command.

Related topics

Get-ProvServiceStatus
Get-ProvDBConnection
Test-ProvDBConnection

Parameters

-DBConnection <String>

Specifies the database connection string to be used by the MachineCreation Service. Passing in $null will clear any existing database connection configured.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

-Force <SwitchParameter>

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th></th>
</tr>
</thead>
</table>
-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | <String> |
| Accept Pipeline Input? | false |

Return Values


The Set-ProvDBConnection command returns an object containing the status of the MachineCreation Service together with extra diagnostics information.

DBUnconfigured

The MachineCreation Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the MachineCreation Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the MachineCreation Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBNewerVersionThanService

The version of the MachineCreation Service currently in use is incompatible with the version of the MachineCreation Service schema on the database. Upgrade the MachineCreation Service to a more recent version.

DBOlderVersionThanService

The version of the MachineCreation Service schema on the database is incompatible with the version of the MachineCreation Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The MachineCreation Service is running and is connected to a database containing a valid schema.

Failed

The MachineCreation Service has failed.

Unknown
The status of the MachineCreation Service cannot be determined.

Notes
If the command fails, the following errors can be returned.

Error Codes
-----------
InvalidDBConnectionString
The database connection string has an invalid format.
DatabaseConnectionDetailsAlreadyConfigured
There was already a database connection configured. After a configuration is set, it can only be set to $null.
PermissionDenied
You do not have permission to execute this command.
AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.
CommunicationError
There was a problem communicating with the remote service.
ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Set-ProvDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Configures a database connection string for the MachineCreation Service.
-------------------------- EXAMPLE 2 --------------------------
c:\PS>Set-ProvDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Configures an invalid database connection string for the MachineCreation Service.
Set-ProvScheme

Oct 10, 2013
Changes the parameter values for a provisioning scheme.

Syntax

Set-ProvScheme [-ProvisioningSchemeName] <String> [-VMCpuCount <Int32>] [-VMMemoryMB <Int32>] [-PassThru] [-
LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-ProvScheme [-ProvisioningSchemeName] <String> -IdentityPoolName <String> [-VMCpuCount <Int32>] [-VMMemoryMB
<Int32>] [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-ProvScheme [-ProvisioningSchemeName] <String> -IdentityPoolUid <Guid> [-VMCpuCount <Int32>] [-VMMemoryMB
<Int32>] [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-ProvScheme -ProvisioningSchemeUid <Guid> -IdentityPoolUid <Guid> [-VMCpuCount <Int32>] [-VMMemoryMB <Int32>]
[-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-ProvScheme -ProvisioningSchemeUid <Guid> -IdentityPoolName <String> [-VMCpuCount <Int32>] [-VMMemoryMB
<Int32>] [-PassThru] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to update the parameters of an existing provisioning scheme.

This allows the following parameters to be updated:

Number of CPUs that will be used for VMs created from the provisioning scheme. Maximum amount of memory that will be
used for VMs created from the provisioning scheme.

To change the name of the provisioning scheme see Rename-ProvScheme.

Related topics

New-ProvScheme
Remove-ProvScheme
Get-ProvScheme
Rename-ProvScheme

Parameters

-ProvisioningSchemeName <String>

The name of the provisioning scheme to be updated.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ProvisioningSchemeName</td>
<td>true</td>
</tr>
</tbody>
</table>

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.2351
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-IdentityPoolName</code></td>
<td>The name of an identity pool to associate with the provisioning scheme, replacing the present one.</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td><code>-IdentityPoolUid</code></td>
<td>The identifier of an identity pool to associate with the provisioning scheme, replacing the present one.</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td><code>-ProvisioningSchemeUid</code></td>
<td>The identifier of the provisioning scheme to be updated.</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td><code>-VMCpuCount</code></td>
<td>The number of processors that virtual machines created from the provisioning scheme should use.</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td><code>-VMMemoryMB</code></td>
<td>The maximum amount of memory that virtual machines created from the provisioning scheme should use.</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>
-**PassThru**<SwitchParameter>

Defines whether or not the command returns a result showing the new state of the updated provisioning scheme.

-**LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-**AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

**Return Values**

Citrix.MachineCreation.Sdk.ProvisioningScheme

This object provides details of the provisioning scheme and contains the following information:

ProvisioningSchemeUid

ProvisioningSchemeName

The name of the provisioning scheme.

CpuCount
The number of processors that VMs will be created with when using this scheme.

MemoryMB

The maximum amount of memory that VMs will be created with when using this scheme.

MasterImageVM

The path within the hosting unit provider to the VM or snapshot of which the scheme is currently using a copy.

MasterImageVMDate

The date and time that the copy of the VM image was made for the scheme.

IdentityPoolUid

The unique identifier of the identity pool (from the ADIdentity PowerShell snap-in) that the scheme uses.

IdentityPoolName

The name of the identity pool (from the ADIdentity PowerShell snap-in) that the scheme uses.

HostingUnitUid

The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme will use.

HostingUnitName

The name of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme will use.

CleanOnBoot

Indicates whether the VMs created are to be reset to a clean state on each boot.

TaskId

The identifier of any current task that is running for the provisioning scheme.

Notes

In the case of failure, the following errors can result.

Error Codes

----------

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-ProvScheme -ProvisioningSchemeName MyScheme -VMCpuCount 2
Updates a provisioning scheme called "MyScheme" to use two processors on the VMs that are created from the provisioning scheme.
Set-ProvSchemeMetadata

Oct 10, 2013

Adds or updates metadata on the given ProvisioningScheme.

Syntax

Set-ProvSchemeMetadata [-ProvisioningSchemeUid] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Set-ProvSchemeMetadata [-ProvisioningSchemeUid] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Set-ProvSchemeMetadata [-ProvisioningSchemeName] <String> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Set-ProvSchemeMetadata [-ProvisioningSchemeName] <String> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Set-ProvSchemeMetadata [-InputObject] <ProvisioningScheme[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Set-ProvSchemeMetadata [-InputObject] <ProvisioningScheme[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

Provides the ability for additional custom data to be stored against given ProvisioningScheme objects.

Related topics

Add-ProvSchemeMetadata

Remove-ProvSchemeMetadata

Parameters

-ProvisioningSchemeUid <Guid>

Id of the ProvisioningScheme

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-ProvisioningSchemeName <String>

Name of the ProvisioningScheme

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |
- **InputObject** `<ProvisioningScheme[]>`

Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Map** `<PSObject>`

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name** `<String>`

Specifies the property name of the metadata to be added. The property must be unique for the ProvisioningScheme specified. The property cannot contain any of the following characters `/,:.*?=<>|[]()"'

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Value** `<String>`

Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId** `<Guid>`

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**-AdminAddress <String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

### Return Values


Set-ProvSchemeMetadata returns a dictionary containing the new (name, value)-pairs.

- **Key <string>**
  Specifies the name of the property.

- **Value <string>**
  Specifies the value for the property.

### Notes

If the command fails, the following errors can be returned.

#### Error Codes

- **InvalidParameterCombination**
  The cmdlet parameters are inconsistent.

- **UnknownObject**
  One of the specified objects was not found.

- **DatabaseError**
  An error occurred in the service while attempting a database operation.

- **DatabaseNotConfigured**
  The operation could not be completed because the database for the service is not configured.

- **DataStoreException**
  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- **PermissionDenied**
  You do not have permission to execute this command.

- **AuthorizationError**
  There was a problem communicating with the Citrix Delegated Administration Service.
ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Set-ProvSchemeMetadata -ProvisioningSchemeUid 4CECC26E-48E1-423F-A1F0-2A06DD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the ProvisioningScheme with the identifier '4CECC26E-48E1-423F-A1F0-2A06DD0805C'.
Set-ProvServiceConfigurationData

Oct 10, 2013
Sets the value for the given key in the service configuration data.

Syntax

Set-ProvServiceConfigurationData [-Name <String>] -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

Provides the ability for additional custom data to be stored for the MachineCreation Service.

Related topics

Remove-ProvServiceConfigurationData
Get-ProvServiceConfigurationData

Parameters

-Name <String>

Specifies the key name of the metadata to be added. The key must be unique.

The Name cannot contain any of the following characters \/:.*?<>|[]''

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-Value <String>

Specifies the value for the name. If the name already exists, its value is updated.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
### -AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**

Citrix.MachineCreation.Sdk.ServiceConfigurationData

Set-ProvServiceConfigurationData returns an object containing the new definition of the configuration.

**Name <string>**

Specifies the name for the item of data.

**Value <string>**

Specifies the value of the data.

**Notes**

In the case of failure the following errors can be produced.

**Error Codes**

----------

**DatabaseError**

An error occurred in the service while attempting a database operation.

**DatabaseNotConfigured**

The operation could not be completed because the database for the service is not configured.

**DataStoreException**

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

**PermissionDenied**

You do not have permission to execute this command.
AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
An error occurred while communicating with the service.

ExceptionThrown
An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> Set-ProvServiceConfigurationData -Name "customProperty1" -Value "value2"

Name                Value
--------            -----  
customProperty1    value2

Set data with a name of ‘customProperty1’ and value of ‘value2’ to the service configuration.
Set-ProvServiceMetadata

Oct 10, 2013
Adds or updates metadata on the given Service.

Syntax

Set-ProvServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-ProvServiceMetadata [-ServiceHostId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-ProvServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-ProvServiceMetadata [-InputObject] <Service[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Allows you to store additional custom data against given Service objects.

Related topics
Remove-ProvServiceMetadata

Parameters

-**ServiceHostId**<Guid>
Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-**InputObject**<Service[]>
Objects to which metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Map**<PSObject>
Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @

---

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.2363
"val1": "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | true (ByValue)

- **Name**<String>

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/:#.*?<>|[]()"

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | false

- **Value**<String>

Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | false

- **LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | false

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default.
Accept Pipeline Input?  false

Return Values
Set-ProvServiceMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>
Specifies the name of the property.

Value <string>
Specifies the value for the property.

Notes
If the command fails, the following errors can be returned.

Error Codes
--------

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Set-ProvServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

Key   Value
---   -----
property value

Add metadata with a name of 'property' and a value of 'value' to the Service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Set-ProvTaskMetadata

Oct 10, 2013
Adds or updates metadata on the given Task.

Syntax

Set-ProvTaskMetadata [-TaskId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Set-ProvTaskMetadata [-TaskId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Set-ProvTaskMetadata [-InputObject] <Task[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Set-ProvTaskMetadata [-InputObject] <Task[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Use this cmdlet to store additional custom data against given Task objects.

Related topics

Add-ProvTaskMetadata
Remove-ProvTaskMetadata
Get-ProvTask
Stop-ProvTask
Remove-ProvTask
Switch-ProvTask

Parameters

-TaskId <Guid>
Id of the Task

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-InputObject <Task[]>

Objects to which the metadata is to be added.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>-Name</strong>&lt;String&gt;</td>
<td>Specifies the property name of the metadata to be added. The property must be unique for the Task specified. The property cannot contain any of the following characters /:.*?=&lt;&gt;</td>
</tr>
<tr>
<td><strong>-Value</strong>&lt;String&gt;</td>
<td>Specifies the value for the property.</td>
</tr>
<tr>
<td><strong>-Map</strong>&lt;PSObject&gt;</td>
<td>Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{&quot;name1&quot; = &quot;val1&quot;; &quot;name2&quot; = &quot;val2&quot;}) or a string dictionary (created with new-object &quot;System.Collections.Generic.Dictionary[String,String]&quot;).</td>
</tr>
<tr>
<td><strong>-LoggingId</strong>&lt;Guid&gt;</td>
<td>Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.</td>
</tr>
</tbody>
</table>
**-AdminAddress**: <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**


Set-ProvTaskMetadata returns a dictionary containing the new (name, value)-pairs.

**Key**: <string>

Specifies the name of the property.

**Value**: <string>

Specifies the value for the property.

**Notes**

If the command fails, the following errors can result.

**Error Codes**

------------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Set-ProvTaskMetadata -TaskId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Task with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Stop-ProvTask

Oct 10, 2013
Stops currently running MachineCreation Service tasks.

Syntax

Stop-ProvTask [-TaskId] <Guid> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters>

Detailed Description

Enables tasks currently running in the MachineCreation Service to be stopped. Once stopped, tasks cannot be restarted.

Related topics

Get-ProvTask
Remove-ProvTask
Switch-ProvTask

Parameters

-TaskId <Guid>

Specifies the identifier for the task to be stopped.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

System.Management.Automation.PSObject Objects containing the TaskId parameter can be piped to the Stop-ProvTask command.

Notes

If the command fails, the following errors can result.

Error Codes

---------

InvalidWorkflow
The specified task could not be found.

InvalidWorkflowHost
The specified task is executing on a different server.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

DatabaseError
There was a problem communicating with the database.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-ProvTask -active $true | Stop-ProvTask
Success
Terminate all tasks currently executing on the MachineCreation Service.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Stop-ProvTask -TaskId bd52e688-e40d-4790-83de-9f7633481454
Success
Terminate the named task executing on the MachineCreation Service.
Switch-ProvTask

Oct 10, 2013
Moves all MachineCreation Service tasks from the current execution host to another.

Syntax
Switch-ProvTask [-Host2] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Enables tasks running within the MachineCreation Service to be moved from one execution host to a different host.

Tasks can only execute on a single host. If the host is removed from a deployment, the tasks cannot continue to execute. These 'orphaned' tasks can be moved to a different host within the deployment so that they can continue to execute. All tasks must be moved; there is no mechanism to move individual tasks. Run the Switch-ProvTask command against the host to which the tasks are to be moved; that is, if you are not running the command directly on the host, use the AdminAddress parameter to specify the host to which tasks will be moved.

Related topics
Get-ProvTask
Stop-ProvTask
Remove-ProvTask

Parameters
-Host2 <String>
Specifies the host from which the tasks should be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can result.

Error Codes

----------

PartialData

Only a subset of the requested data was returned.

NoOp

The operation was successful but had no effect.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.
ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Switch-ProvTask -Host CRASHED
Success
Transferring the execution of tasks that had been executing on the MachineCreation Service instance on host CRASHED to the local instance.
Test-ProvDBConnection

Oct 10, 2013
Tests a database connection for the MachineCreation Service.

Syntax

Test-ProvDBConnection [-DBConnection] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters]

Detailed Description

Tests a connection to the database in which the MachineCreation Service can store its state. The service will attempt to connect to the database without affecting the current connection to the database.

You do not have to clear the connection to use this command.

Related topics

Get-ProvServiceStatus
Get-ProvDBConnection
Set-ProvDBConnection

Parameters

- **DBConnection**<String>

Specifies the database connection string to be tested by the MachineCreation Service.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
The Test-ProvDBConnection command returns an object containing the status of the MachineCreation Service if the connection string of the specified data store were to be set to the string being tested, together with extra diagnostics information for the specified connection string.

**DBRejectedConnection**

The database rejected the logon attempt from the MachineCreation Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**

The expected stored procedures are missing from the database. This may be because the MachineCreation Service schema has not been added to the database.

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**

The version of the MachineCreation Service currently in use is incompatible with the version of the MachineCreation Service schema on the database. Upgrade the MachineCreation Service to a more recent version.

**DBOlderVersionThanService**

The version of the MachineCreation Service schema on the database is incompatible with the version of the MachineCreation Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**

A database schema upgrade is currently in progress.

**OK**

The Set-ProvDBConnection command would succeed if it were executed with the supplied connection string.

**Failed**

The MachineCreation Service has failed.

**Unknown**

The status of the MachineCreation Service cannot be determined.

**Notes**

If the command fails, the following errors can be returned.

**Error Codes**

------------

**InvalidDBConnectionString**

The database connection string has an invalid format.

**PermissionDenied**

You do not have permission to execute this command.

**AuthorizationError**

There was a problem communicating with the Citrix Delegated Administration Service.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Test-ProvDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Tests a database connection string for the MachineCreation Service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Test-ProvDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Tests an invalid database connection string for the MachineCreation Service.
Test-ProvSchemeNameAvailable

Oct 10, 2013
Checks to ensure that the proposed name for a provisioning scheme is unused.

Syntax

Test-ProvSchemeNameAvailable -ProvisioningSchemeName <String[]> [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Checks to ensure that the proposed name for a provisioning scheme is unused. This check is done without regard for scoping of existing provisioning schemes, so the names of inaccessible schemes are also checked.

Related topics

New-ProvScheme

Rename-ProvScheme

Parameters

-ProvisioningSchemeName <String[]>

The name or names of the provisioning scheme(s) to be tested.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

NameAvailability

Pairs of name and the availability of the name

Notes

In the case of failure, the following errors can result.
Error Codes

-----------

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

Test-ProvSchemeNameAvailable -ProvisioningSchemeName $NewSchemeName
This tests whether the value of $NewSchemeName is unique or not, and can be used to create a new provisioning scheme or rename an existing one without failing. True is returned if the name is good.
Unlock-ProvScheme

Oct 10, 2013
Unlocks a Provisioning Scheme.

Syntax

Unlock-ProvScheme [-ProvisioningSchemeName] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Unlock-ProvScheme -ProvisioningSchemeUid <Guid> [-LoggingId <Guid>] [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

Provides the ability to unlock a provisioning scheme that has the Id of a failed Task still associated with it. This allows another long-running task to operate on that scheme. The cmdlet will not unlock a scheme with a task still marked as being active. Use Stop-ProvTask (or, if the task is registered with a failed server, Switch-ProvTask and Stop-ProvTask) to stop any active task first.

Related topics

Get-ProvScheme
Stop-ProvTask

Parameters

-ProvisioningSchemeName <String>
The name of the provisioning scheme.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByPropertyName) |

-ProvisioningSchemeUid <Guid>
The unique identifier of the provisioning scheme.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | false |

-LoggingId <Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create
high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | LocalHost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

**Notes**

In the case of failure, the following errors can result.

**Error Codes**

-------------

ProvisioningSchemeNotFound
The specified provisioning scheme could not be located.

NoOp
The specified provisioning scheme had no task object associated.

TaskActive
The task object associated with the provisioning scheme is still active.

DatabaseError
An error occurred in the service while attempting a database operation.

CouldNotQueryDatabase
The query required to get the database was not defined.

CommunicationError
An error occurred while communicating with the service.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Unlock-ProvScheme -provisioningSchemeName MyScheme
Unlocks the provisioning scheme 'MyScheme'.
Unlock-ProvVM

Oct 10, 2013
Unlocks a VM.

Syntax

Unlock-ProvVM [-VMID] <String[]> -ProvisioningSchemeName <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Unlock-ProvVM [-VMID] <String[]> -ProvisioningSchemeUid <Guid> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to unlock a virtual machine.

Related topics

Get-ProvVM
Lock-ProvVM

Parameters

-VMID <String[]>

The virtual machine Id (hypervisor context)

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByPropertyName) |

-ProvisioningSchemeName <String>

The name of the provisioning scheme.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | true (ByPropertyName) |

-ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme.

| Required? | true |
|
- **LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller that the PowerShell snap-in connects to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>LocalHost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine You can pipe an object containing a parameter called 'VMId' and 'ProvisioningSchemeName' to Lock-ProvVM

Notes

In the case of failure, the following errors can result.

Error Codes

----------

VMDoesNotExist

The specified VM cannot be located.

VMAReadyUnLocked

The VM is already unlocked.

VMDoesNotExistForProvisioningScheme

The specified VM does exist in the hypervisor, but is not part of the specified provisioning scheme.
DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError
An error occurred while communicating with the service.

PermissionDenied
The user does not have administrative rights to perform this operation.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error

ExceptionThrown
An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
C:\PS>Unlock-ProvVM -provisioningSchemeName MyScheme -VMId bc79802c-ba6e-8de8-99e9-4c35d7ad24b4
Unlocks the VM with the Id 'bc79802c-ba6e-8de8-99e9-4c35d7ad24b4' in the provisioning scheme 'MyScheme'.

-------------------------- EXAMPLE 2 --------------------------
C:\PS>Get-ProvVM -provisioningSchemeName MyScheme | Unlock-ProvVM
Unlocks all the VMs in the provisioning scheme 'MyScheme'.

-------------------------- EXAMPLE 3 --------------------------
C:\PS>Get-ProvVM -Locked $True | Unlock-ProvVM
Unlocks all the VMs that are currently locked.
## Overview

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MonitorSnapin</td>
<td>The Monitor Service PowerShell snap-in provides administrative functions for the Monitor Service.</td>
</tr>
<tr>
<td>Monitor Filtering</td>
<td>Describes the common filtering options for XenDesktop cmdlets.</td>
</tr>
</tbody>
</table>

### Cmdlets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get-MonitorConfiguration</td>
<td>Gets the configuration settings currently being used by the Monitor Service.</td>
</tr>
<tr>
<td>Get-MonitorDataStore</td>
<td>Gets details for each of the Monitor data stores.</td>
</tr>
<tr>
<td>Get-MonitorDBConnection</td>
<td>Gets the database string for the specified data store used by the Monitor Service.</td>
</tr>
<tr>
<td>Get-MonitorDBSchema</td>
<td>Gets a script that creates the Monitor Service database schema for the specified data store.</td>
</tr>
<tr>
<td>Get-MonitorDBVersionChangeScript</td>
<td>Gets a script that updates the Monitor Service database schema.</td>
</tr>
<tr>
<td>Get-MonitorInstalledDBVersion</td>
<td>Gets a list of all available database schema versions for the Monitor Service.</td>
</tr>
<tr>
<td>Get-MonitorService</td>
<td>Gets the service record entries for the Monitor Service.</td>
</tr>
<tr>
<td>Get-MonitorServiceAddedCapability</td>
<td>Gets any added capabilities for the Monitor Service on the controller.</td>
</tr>
<tr>
<td>Get-MonitorServiceInstance</td>
<td>Gets the service instance entries for the Monitor Service.</td>
</tr>
<tr>
<td>Get-MonitorServiceStatus</td>
<td>Gets the current status of the Monitor Service on the controller.</td>
</tr>
<tr>
<td>Remove-MonitorServiceMetadata</td>
<td>Removes metadata from the given Service.</td>
</tr>
<tr>
<td>Reset-MonitorDataStore</td>
<td>Refreshes the database string currently being used by the Monitor service.</td>
</tr>
<tr>
<td>Reset-MonitorServiceGroupMembership</td>
<td>Reloads the access permissions and configuration service locations for the Monitor Service.</td>
</tr>
<tr>
<td>Set-MonitorConfiguration</td>
<td>Sets the database string currently being used by the Monitor Service.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>Set-MonitorDBConnection</td>
<td>Configures a database connection for the Monitor Service.</td>
</tr>
<tr>
<td>Set-MonitorServiceMetadata</td>
<td>Adds or updates metadata on the given Service.</td>
</tr>
<tr>
<td>Test-MonitorDBConnection</td>
<td>Tests a database connection for the Monitor Service.</td>
</tr>
</tbody>
</table>
about_MonitorMonitorSnapin

Oct 10, 2013

TOPIC

about_MonitorMonitorSnapin

SHORT DESCRIPTION

The Monitor Service PowerShell snap-in provides administrative functions for the Monitor Service.

COMMAND PREFIX

All commands in this snap-in have the noun prefixed with 'Monitor'.

LONG DESCRIPTION

The Monitor Service PowerShell snap-in enables both local and remote administration of the Monitor service. It provides facilities to query the Monitor service configuration settings and to modify those settings. It also provides the standard set of XenDesktop service cmdlets.
about_Monitor_Filtering

Oct 10, 2013

TOPIC

XenDesktop - Advanced Dataset Filtering

SHORT DESCRIPTION

Describes the common filtering options for XenDesktop cmdlets.

LONG DESCRIPTION

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get-cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

```
Get-<Noun> | Where { $_.Size = 'Small' } | Sort 'Date' | Select -First 10
```

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get-cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

```
Get-<Noun> -Size 'Small' -SortBy 'Date' -MaxRecordCount 10
```

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

- **MaxRecordCount <int>**

  Specifies the maximum number of results to return.

  For example, to return only the first nine results use:

  ```
  Get-<Noun> -MaxRecordCount 9
  ```

  If not specified, only the first 250 records are returned, and if more are available, a warning is produced:
WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

To retrieve all records, specify a large number for -MaxRecordCount. As the value is an integer, you can use the following:

```
Get-<Noun> -MaxRecordCount [int]:MaxValue
```

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
...
```

```
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<< -MaxRecordCount 9 -ReturnTotalRecordCount
   + CategoryInfo : OperationStopped: (.) [Get-<Noun>], PartialDataException
   + FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

```
$count = $error[0].TotalAvailableResultCount
```

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.
Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter 'Name -like "High**" -or (Priority -eq 1 -and Severity -ge 2)'

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

EXAMPLES

Filtering by strings performs a case-insensitive wildcard match.
Separate parameters are combined with an implicit -and operator.
Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company "citrix" -Product '[X]EN''
Get-<Noun> -Product "Xen" -Company "CITRIX"
Get-<Noun> -Filter { Company -eq 'Citrix' -and Product -like 'Xen' }
See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```powershell
Get-<Noun> -Company "Abc[*]"                 # Matches Abc*
Get-<Noun> -Company "Abc**"                  # Matches Abc*
Get-<Noun> -Filter { Company -eq "Abc**" }  # Matches Abc*
Get-<Noun> -Filter { Company -eq "A""B""C" } # Matches A"B"C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```powershell
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

```powershell
Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```powershell
Get-<Noun> -Filter 'Enabled'      # Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled' # Equivalent to 'Enabled -eq $false'
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```powershell
Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:
By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

```powershell
Get-INFN> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z" }
$datetime = [DateTime]"2010-08-23T12:30:00.0Z"
Get-INFN> -Filter { StartTime -ge $datetime }
$datetime = (Get-Date).AddDays(-1)
Get-INFN> -Filter { StartTime -ge $datetime }
```

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

```powershell
Get-INFN> -Filter { StartTime -ge '-2' } # Two days ago
Get-INFN> -Filter { StartTime -ge '-1:30' } # Hour and a half ago
Get-INFN> -Filter { StartTime -ge '-0:0:30' } # 30 seconds ago
```

**ARRAY PROPERTIES**

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings. Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

```powershell
Get-INFN> -User Fred*
Get-INFN> -Filter { User -like "Fred*" }
Get-INFN> -Filter { Users -contains "Fred*" }
```

You can also use the singular form with -Filter to search using other operators:

```powershell
# Match if any user in the list is called "Frederick"
Get-INFN> -Filter { User -eq "Frederick" }
# Match if any user in the list has a name alphabetically below 'F'
Get-INFN> -Filter { User -lt 'F' }
```

**COMPLEX EXPRESSIONS**
When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

```
Get-<Noun> -Filter { Shape -eq 'Circle' -or Shape -eq 'Square' }  
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }  
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }
```

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

```
Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue' -and Shape -eq 'Circle') }  
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue' -and Shape -eq 'Circle') }
```

**PAGING**

The simplest way to page through data is to use the -Skip and -MaxRecordCount parameters. So, to read the first three pages of data with 10 records per page, use:

```
Get-<Noun> -Skip 0 -MaxRecordCount 10 <other filtering criteria>  
Get-<Noun> -Skip 10 -MaxRecordCount 10 <other filtering criteria>  
Get-<Noun> -Skip 20 -MaxRecordCount 10 <other filtering criteria>
```

You must include the same filtering criteria on each call, and ensure that the data is sorted consistently.

The above approach is often acceptable, but as each call performs an independent query, data changes can result in records being skipped or appearing twice. One approach to improve this is to sort by a unique id field and then start the search for the next page at the unique id after the last unique id of the previous page. For example:

```
# Get the first page
Get-<Noun> -MaxRecordCount 10 -SortBy SerialNumber

Serializable ...  
--------  ---
A120004
A120007
... 7 other records ...
A120900

# Get the next page
Get-<Noun> -MaxRecordCount 10 -Filter { FirstName -gt 'A120900' }

Serializable ...
FILTER SYNTAX DEFINITION

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= "{" <ComponentList> "}"

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> | <Component>

<Component> ::= <NotOperator> <Factor> | <Factor>

<Factor> ::= "(" <ComponentList> ")" | <PropertyName> <ComparisonOperator> <Value> | <PropertyName>

<AndOrOperator> ::= "-and" | "-or"

<NotOperator> ::= "-not" | "!

<ComparisonOperator> ::= "-eq" | "-ne" | "-le" | "-lt" | "-ge" | "-gt" | "-like" | "-notlike" | "-contains" | "-notcontains" | "-in" | "-notin"

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> | <scalar variable> | <array variable> | "$null" | "$true" | "$false"
Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30" means 1 hour and 30 minutes ago.
Get-MonitorConfiguration

Oct 10, 2013

Gets the configuration settings currently being used by the Monitor Service.

Syntax


Detailed Description

Returns the configuration currently being used by the Monitor Service.

A site database connection must be configured for this command to be used.

Related topics

Set-MonitorConfiguration

Parameters

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Get-MonitorConfiguration returns the configuration settings.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-MonitorConfiguration
Gets the current configuration for the Monitor Service.
Get-MonitorDataStore

Oct 10, 2013
Gets details for each of the Monitor data stores.

Syntax

Get-MonitorDataStore [-AdminAddress <String>] [CommonParameters]

Detailed Description

Returns an object for each of the Monitor data stores describing the connection string, data store name, db type, provider, schema name, and DB status.

A database connection must be configured in order for this command to be used if the service has a secondary data store. This is not required for the site data store.

Related topics

Reset-MonitorDataStore

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

Citrix.Monitor.Sdk.DataStoreConfiguration

An object describing the connection string, data store name, database type, provider, schema name and database status.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-MonitorDataStore

ConnectionString : Server=\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True
DataStore        : Site
DatabaseType     : SqlServer
Provider         : MSSQL
SchemaName       : MonitorSiteSchema
Status           : OK

ConnectionString :
DataStore        : Secondary
DatabaseType     : SqlServer
Provider         : MSSQL
SchemaName       : MonitorSecondarySchema
Status           : DBUnconfigured

Get the database connection string for the Monitor Service.
Get-MonitorDBConnection

Oct 10, 2013
Gets the database string for the specified data store used by the Monitor Service.

Syntax
Get-MonitorDBConnection [-DataStore <String>] [-AdminAddress <String>] [CommonParameters]

Detailed Description
Returns the database connection string for the specified data store.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

Related topics
Get-MonitorServiceStatus
Get-MonitorDataStore
Set-MonitorDBConnection
Test-MonitorDBConnection

Parameters

-DataStore <String>
Specifies the logical name of the data store for the Monitor Service. Can be either be 'Site' or the logical name of the secondary data store.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Site</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

https://docs.citrix.com  © 1999-2017 Citrix Systems, Inc. All rights reserved.  p.2403
systemstring

The database connection string configured for the Monitor Service.

Notes
If the command fails, the following errors can be returned.

Error Codes
-------------

NoDBConnections
The database connection string for the Monitor Service has not been specified.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-MonitorDBConnection

Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True
Get the database connection string for the Monitor Service.
Get-MonitorDBSchema

Oct 10, 2013
Gets a script that creates the Monitor Service database schema for the specified data store.

Syntax


Detailed Description

Gets SQL scripts that can be used to create a new Monitor Service database schema, add a new Monitor Service to an existing site, remove a Monitor Service from a site, or create a database server logon for a Monitor Service. If no Sid parameter is provided, the scripts obtained relate to the currently selected Monitor Service instance, otherwise the scripts relate to Monitor Service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied. The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to a Monitor SDK cmdlet. The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SMDCMD mode'. The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to Monitor Service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user

If ScriptType is Evict, the returned script contains:

- Removal of Monitor Service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

If the service uses two data stores they can exist in the same database. You do not need to configure a database before using this command.

Related topics

- Get-MonitorDataStore
- Set-MonitorDBConnection
Parameters

-DatabaseName <String>

Specifies the name of the database for which the schema will be generated.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

-ServiceGroupName <String>

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the Monitor services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

| Required? | false |
| Default Value |
| Accept Pipeline Input? | false |

-ScriptType <ScriptTypes>

Specifies the type of database script returned. Available script types are:

Database

Returns a full database script that can be used to create a database schema for the Monitor Service in a database instance that does not already contain a schema for this service. The DatabaseName and ServiceGroupName parameters must be specified to create a script of this type.

Instance

Returns a permissions script that can be used to add further Monitor services to an existing database instance that already contains the full Monitor service schema, associating the services to the Service Group. The Sid parameter can optionally be specified to create a script of this type.

Login

Returns a database logon script that can be used to add the required logon accounts to an existing database instance that contains the Monitor Service schema. This is used primarily when creating a mirrored database environment. The DatabaseName parameter must be specified to create a script of this type.

Evict

Returns a script that can be used to remove the specified Monitor Service from the database entirely. The DatabaseName and Sid parameters must be specified to create a script of this type.
- **LocalDatabase**<SwitchParameter>

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for Monitor services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Sid**<String>

Specifies the SID of the controller on which the Monitor Service instance to remove from the database is running.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **DataStore**<String>

Specifies the logical name of the data store for the Monitor Service. Can be either be 'Site' or the logical name of the secondary data store.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Site</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

**System.String**

A string containing the required SQL script for application to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will
Create the database schema.
Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:
Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:
Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned.

Error Codes
-----------

GetSchemasFailed
The database schema could not be found.

ActiveDirectoryAccountResolutionFailed
The specified Active Directory account or Group could not be found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-MonitorDBSchema -DatabaseName MyDB -ServiceGroupName MyServiceGroup > c:\MonitorSchema.sql
Get the full database schema for site data store of the Monitor Service and copy it to a file called 'c:\MonitorSchema.sql'.
This script can then be used to create the schema in a pre-existing database named 'MyDB' that does not already contain a Monitor Service site schema.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Get-MonitorDBSchema -DatabaseName MyDB -scriptType Login > c:\MonitorLogins.sql
Get the logon scripts for the Monitor Service.

-------------------------- EXAMPLE 3 --------------------------
c:\PS>Get-MonitorDBSchema -DatabaseName MyDB -ServiceGroupName MyServiceGroup -DataStore Secondary > c:\MonitorSchema.sql
Get the full database schema for the secondary data store of the Monitor Service and copy it to a file called 'c:\MonitorSecondarySchema.sql'.
This script can then be used to create the schema in a pre-existing database named 'MyDB' that does not already contain a Monitor Service secondary schema.
Get-MonitorDBVersionChangeScript

Oct 10, 2013
Gets a script that updates the Monitor Service database schema.

Syntax

Get-MonitorDBVersionChangeScript -DatabaseName <String> -TargetVersion <Version> [-DataStore <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns a database script that can be used to upgrade or downgrade the site or secondary schema for the Monitor Service from the current schema version to a different version.

Related topics

Get-MonitorInstalledDBVersion

Parameters

- **DatabaseName**:<String>

  Specifies the name of the database instance to which the update applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **TargetVersion**:<Version>

  Specifies the version of the database you want to update to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **DataStore**:<String>

  Specifies the logical name of the data store for the Monitor Service. Can be either be 'Site' or the logical name of the secondary data store.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Site</td>
<td>false</td>
</tr>
</tbody>
</table>
- **AdminAddress**: <String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

  | Required? | false |
  | Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
  | Accept Pipeline Input? | false |

**Return Values**

System.Management.Automation.PSObject

A PSObject containing the required SQL script for application to a database.

**Notes**

The PSObject returned by this cmdlet contains the following properties:

--- Script
The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

--- NeedExclusiveAccess
Indicates whether all services in the service group must be shut down during the update or not.

--- CanUndo
Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any Monitor services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-MonitorServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned.

**Error Codes**

----------

**NoOp**

The operation was successful but had no effect.

**NoDBConnections**

The database connection string for the Monitor Service has not been specified.
DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $update = Get-MonitorDBVersionChangeScript -DatabaseName MyDb -TargetVersion 1.0.75.0
C:\PS> $update.Script > update_75.sql

Gets an SQL update script to update the current schema to version 1.0.75.0. The resulting update_75.sql script is suitable for direct use with the SQL Server SQLCMD utility.
Get-MonitorInstalledDBVersion

Oct 10, 2013
Gets a list of all available database schema versions for the Monitor Service.

Syntax


Detailed Description

Returns the current version of the Monitor Service database schema, if no flags are set, otherwise returns versions for which upgrade or downgrade scripts are available and have been stored in the database.

Related topics

Parameters

- **Upgrade**<SwitchParameter>

  Specifies that only schema versions to which the current database version can be updated should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Downgrade**<SwitchParameter>

  Specifies that only schema versions to which the current database version can be reverted should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **DataStore**<String>

  Specifies the database connection logical name the schema script should be returned for. The parameter is optional.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>


Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.Version

The Get-MonitorInstalledDbVersion command returns objects containing the new definition of the Monitor Service database schema version.

Major <Integer>
Minor <Integer>
Build <Integer>
Revision <Integer>

Notes

If the command fails, the following errors can be returned.

Error Codes

--------

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the Monitor Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-MonitorInstalledDBVersion

<table>
<thead>
<tr>
<th>Major</th>
<th>Minor</th>
<th>Build</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Get the currently installed version of the Monitor Service database schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-MonitorInstalledDBVersion -Upgrade

<table>
<thead>
<tr>
<th>Major</th>
<th>Minor</th>
<th>Build</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Get the versions of the Monitor Service database schema for which upgrade scripts are supplied.
Get-MonitorService

Oct 10, 2013
Gets the service record entries for the Monitor Service.

Syntax


Detailed Description

Returns instances of the Monitor Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Related topics

Parameters

- **-Property <String[]>**
  Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

- **-ReturnTotalRecordCount <SwitchParameter>**
  When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Monitor_Filtering for details.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

- **-MaxRecordCount <Int32>**
  Specifies the maximum number of records to return.

<p>| Required? | false |</p>
<table>
<thead>
<tr>
<th>Default Value</th>
<th>250</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Skip<Int32>**

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | false |

**-SortBy<String>**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |
| Accept Pipeline Input? | false |

**-Filter<String>**

Gets records that match a PowerShell-style filter expression. See about_Monitor_Filtering for details.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |
Citrix.Monitor.Sdk.Service

The Get-MonitorServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>
Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

---------

UnknownObject
One of the specified objects was not found.

PartialData
Only a subset of the available data was returned.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase
The query required to get the database was not defined.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.
Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-MonitorService

Uid  : 1
ServiceHostId  : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName       : MyServer.company.com
MachineName   : MYSERVER
CurrentState  : On
LastStartTime : 04/04/2011 15:25:38
LastActivityTime : 04/04/2011 15:33:39
OSType        : Win32NT
OSVersion     : 6.1.7600.0
ServiceVersion: 5.1.0.0
DatabaseUserName : NT AUTHORITY\NETWORK SERVICE
SID           : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {MySiteService1, MySiteService2...}

Get all the instances of the Monitor Service running in the current service group.
Get-MonitorServiceAddedCapability

Oct 10, 2013

Gets any added capabilities for the Monitor Service on the controller.

Syntax

Get-MonitorServiceAddedCapability [-AdminAddress <String>] [CommonParameters]

Detailed Description

Enables updates to the Monitor Service on the controller to be detected.

You do not need to configure a database connection before using this command.

Related topics

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

System.String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-MonitorServiceAddedCapability
Get the added capabilities of the Monitor Service.
Get-MonitorServiceInstance

Oct 10, 2013
Gets the service instance entries for the Monitor Service.

Syntax
Get-MonitorServiceInstance [-AdminAddress <String>] [CommonParameters]

Detailed Description
Returns service interfaces published by the instance of the Monitor Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Related topics
Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
Citrix.Monitor.Sdk.ServiceInstance

The Get-MonitorServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>
Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>
Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>
Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.
ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Monitor.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

- SDK - for PowerShell operations
- InterService - for operations between different services
- Peer - for communications between services of the same type

Metadata <Citrix.Monitor.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

---------

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-MonitorServiceInstance

Address            : http://MyServer.com:80/Citrix/Monitor
Binding            : wcf_HTTP_kerb
InterfaceType      : SDK
Metadata           :
MetadataMap        :
ServiceAccount     : ENG\MyAccount$
ServiceAccountSid  : S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName   : MyServiceGroup
ServiceGroupUid    : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceUid : 00000000-0000-0000-0000-000000000000
ServiceType        : Monitor
Version            : 1

Address            : http://MyServer.com:80/Citrix/Monitor/IServiceApi
Binding            : wcf_HTTP_kerb
InterfaceType      : InterService
Metadata           :
MetadataMap        :
ServiceAccount     : ENG\MyAccount
ServiceAccountSid  : S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName : MyServiceGroup
ServiceGroupUid  : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceUid : 00000000-0000-0000-0000-000000000000
ServiceType     : Monitor
Version         : 1

Get all instances of the Monitor Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.
Get-MonitorServiceStatus

Oct 10, 2013
Gets the current status of the Monitor Service on the controller.

Syntax

Get-MonitorServiceStatus [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Enables the status of the Monitor Service on the controller to be monitored. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured.

Related topics

Get-MonitorDataStore
Set-MonitorDBConnection
Test-MonitorDBConnection
Get-MonitorDBConnection
Get-MonitorDBSchema

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


The Get-MonitorServiceStatus command returns an object containing the status of the Monitor Service together with extra diagnostics information.

DBUnconfigured

The Monitor Service does not have a database connection configured.

DBRejectedConnection
The database rejected the logon attempt from the Monitor Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the Monitor Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBNewerVersionThanService

The version of the Monitor Service currently in use is incompatible with the version of the Monitor Service schema on the database. Upgrade the Monitor Service to a more recent version.

DBOlderVersionThanService

The version of the Monitor Service schema on the database is incompatible with the version of the Monitor Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Monitor Service is running and is connected to a database containing a valid schema.

Failed

The Monitor Service has failed.

Unknown

(0) The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-MonitorServiceStatus

DBUnconfigured
Get the current status of the Monitor Service.
Remove-MonitorServiceMetadata

Oct 10, 2013
Removes metadata from the given Service.

Syntax

Remove-MonitorServiceMetadata [-ServiceHostId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-MonitorServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-MonitorServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-MonitorServiceMetadata [-InputObject] <Service[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given Service.

Related topics

Set-MonitorServiceMetadata

Parameters

-**ServiceHostId**<Guid>

Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-**InputObject**<Service[]>

Objects to which metadata is to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-**Name**<String>
The metadata property to remove.

<table>
<thead>
<tr>
<th>Property</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Map&lt;PSObject&gt;</td>
<td>Required?</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-LoggingId&lt;Guid&gt;</td>
<td>Required?</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>Default Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-AdminAddress&lt;String&gt;</td>
<td>Required?</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Accept Pipeline Input?</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

Error Codes

----------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

\c:\PS>Get-MonitorService | % { Remove-MonitorServiceMetadata -Map $_.MetadataMap }
Remove all metadata from all Service objects.
Reset-MonitorDataStore

Oct 10, 2013
Refreshes the database string currently being used by the Monitor service.

Syntax

Reset-MonitorDataStore [-DataStore <String>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns the string for the database connection currently being used by the Monitor Service. Can only be called for secondary data stores.

There is no requirement for a database connection to be configured in order for this command to be used.

Related topics

Get-MonitorDataStore

Parameters

-DataStore <String>

Specifies the database connection logical name to be used by the Monitor Service. Can be either be 'Site' or the logical name of the secondary data store. Specifying the site data store will display an error because this operation is not supported for site data stores.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Site</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
<tr>
<td>Required?</td>
<td>false</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
EXAMPLE 1

c:\PS>Reset-MonitorDataStore -DataStore Secondary
OK
Refresh the database connection string for the Monitor Service.
Reset-MonitorServiceGroupMembership

Oct 10, 2013
Reloads the access permissions and configuration service locations for the Monitor Service.

Syntax

```
```

Detailed Description

Enables you to reload Monitor Service access permissions and configuration service locations. The Reset-MonitorServiceGroupMembership command must be run on at least one instance of the service type (Monitor) after installation and registration with the configuration service. Without this operation, the Monitor services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-MonitorServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Related topics

Parameters

- **-ConfigServiceInstance** <ServiceInstance[]>

  Specifies the configuration service instance object that represents the service instance for the type 'InterService' that references a configuration service for the deployment.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId** <Guid>

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress** <String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Input Type
Citrix.Monitor.Sdk.ServiceInstance[] Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-MonitorServiceGroupMembership command.

Notes
If the command fails, the following errors can be returned.

Error Codes
--------

NoSuitableServiceInstance
None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DatabaseException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples
--------------- EXAMPLE 1 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-MonitorServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

--------------- EXAMPLE 2 --------------------------
c:\PS>Get-ConfigRegisteredServiceInstance -ServiceType Config -AdminAddress OtherServer.example.com | Reset-MonitorServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.
Set-MonitorConfiguration

Sets the database string currently being used by the Monitor Service.

Syntax


Detailed Description

Sets the string for the database connection currently being used by the Monitor Service.

A database connection need not be configured for this command to be used.

Related topics

Get-MonitorConfiguration

Parameters

-GroomSessionsRetentionDays <Int32>

Determines how many days to keep Session and Connection records after the Session is terminated.

| Required? | false |
| Default Value | 7 for non-platinum, 90 for platinum. |
| Accept Pipeline Input? | false |

-GroomFailuresRetentionDays <Int32>

Determines how many days to keep MachineFailureLog and ConnectionFailureLog records after these are created.

| Required? | false |
| Default Value | 7 for non-platinum, 90 for platinum. |
| Accept Pipeline Input? | false |

-GroomLoadIndexesRetentionDays <Int32>

Determines how many days to keep LoadIndex records after these are created.

| Required? | false |
| Default Value | 7 for non-platinum, 90 for platinum. |
| Accept Pipeline Input? | false |
**-GroomDeletedRetentionDays**<Int32>

Determines how many days to keep Machine, Catalog, DesktopGroup and Hypervisor entities around that have a LifecycleState of ‘Deleted’. This also deletes any related Session, Connection, Summary, Failure or LoadIndex records.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>7 for non-platinum, 90 for platinum.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-GroomSummariesRetentionDays**<Int32>

Determines how many days to keep DesktopGroupSummary, FailureLogSummary and LoadIndexSummary records at the daily granularity.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>7 for non-platinum, 90 for platinum.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-DataCollectionEnabled**<Boolean>

Starts / stops data collection. Stopping data collection turns off polling, and does not persist operational event data to the database.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>True</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-FullPollStartHour**<Int32>

Hour of day when Full Poll should begin.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-ResolutionPollTimeHours**<Int32>

Start time for the Resolution Poll worker.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-SyncPollTimeHours</td>
<td>Int32</td>
<td>Start time for Sync Poll worker.</td>
</tr>
<tr>
<td>-DetailedSqlOutputEnabled</td>
<td>Boolean</td>
<td>Determines if the SqlLog should be enabled to send SQL statements to the CDF Trace.</td>
</tr>
<tr>
<td>-LoggingId</td>
<td>Guid</td>
<td>Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.</td>
</tr>
<tr>
<td>-AdminAddress</td>
<td>String</td>
<td>Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.</td>
</tr>
</tbody>
</table>

For more information, please visit the Citrix documentation at https://docs.citrix.com
Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS>Set-MonitorConfiguration -GroomSessionsRetentionDays 5 -GroomFailuresRetentionDays 4 ...
Updates the settings in the site database with the newly specified values.
Set-MonitorDBConnection

Oct 10, 2013
Configures a database connection for the Monitor Service.

Syntax

Set-MonitorDBConnection [-DBConnection] <String> [-Force] [-LoggingId <Guid>] [-AdminAddress <String>] [-DataStore <String>] [<CommonParameters>]

Detailed Description

Configures a connection to a database in which the Monitor Service can store its state. The service will attempt to connect and start using the database immediately after the connection is configured. The database connection string is updated to the specified value regardless of whether it is valid or not. Specifying an invalid connection string prevents a service from functioning until the error is corrected.

After a connection is configured, you cannot alter it without first clearing it (by setting the connection to $null).

You do not need to configure a database connection to use this command.

Related topics

Get-MonitorServiceStatus
Get-MonitorDBConnection
Test-MonitorDBConnection

Parameters

-**DBConnection**<String>

Specifies the database connection string to be used by the Monitor Service. Passing in $null will clear any existing database connection configured.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

-**Force**<SwitchParameter>

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
Accept Pipeline Input? | false
---|---

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-DataStore**<String>

Specifies the logical name of the data store for the Monitor Service. Can be either be 'Site' or the logical name of the secondary data store.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Site</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**


The `Set-MonitorDBConnection` command returns an object containing the status of the Monitor Service together with extra diagnostics information.

**DBUnconfigured**

The Monitor Service does not have a database connection configured.

**DBRejectedConnection**

The database rejected the logon attempt from the Monitor Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**

The expected stored procedures are missing from the database. This may be because the Monitor Service schema has not been added to the database.

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**

The version of the Monitor Service currently in use is incompatible with the version of the Monitor Service schema on the database. Upgrade the Monitor Service to a more recent version.

**DBOlderVersionThanService**

The version of the Monitor Service schema on the database is incompatible with the version of the Monitor Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**

A database schema upgrade is currently in progress.

**OK**
The Monitor Service is running and is connected to a database containing a valid schema.

Failed

The Monitor Service has failed.

Unknown

The status of the Monitor Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

-----------

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured. After a configuration is set, it can only be set to $null.

DatabaseError

An error occurred in the service while attempting a database operation.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Set-MonitorDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"

OK

Configures a database connection string for the Monitor Service.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Set-MonitorDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Configures an invalid database connection string for the Monitor Service.
Set-MonitorServiceMetadata

Oct 10, 2013

Adds or updates metadata on the given Service.

Syntax

Set-MonitorServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-MonitorServiceMetadata [-ServiceHostId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-MonitorServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Allows you to store additional custom data against given Service objects.

Related topics

Remove-MonitorServiceMetadata

Parameters

-ServiceHostId<Guid>

Id of the Service

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-Input Object<Service[]>

Objects to which metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Map<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").
-Name <String>

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/:#.*=?<>|[]"'

-Value <String>

Specifies the value for the property.

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

Return Values
Set-MonitorServiceMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>
Specifies the name of the property.

Value <string>
Specifies the value for the property.

Notes
If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.
ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Set-MonitorServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Test-MonitorDBConnection

Oct 10, 2013
Tests a database connection for the Monitor Service.

Syntax
Test-MonitorDBConnection [-DBConnection] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [-DataStore <String>] [<CommonParameters>]

Detailed Description
Tests a connection to the database in which the Monitor Service can store its state. The service will attempt to connect to the database without affecting the current connection to the database.
You do not have to clear the connection to use this command.

Related topics
Get-MonitorServiceStatus
Get-MonitorDBConnection
Set-MonitorDBConnection

Parameters
-DBConnection<String>
Specifies the database connection string to be tested by the Monitor Service.

| Required? | true |
| Default Value |  |
| Accept Pipeline Input? | false |

-LoggingId<Guid>
Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

| Required? | false |
| Default Value |  |
| Accept Pipeline Input? | false |

-AdminAddress<String>
Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

-DataStore<String>
Specifies the logical name of the data store for the Monitor Service. Can be either be 'Site' or the logical name of the secondary data store.
The Test-MonitorDBConnection command returns an object containing the status of the Monitor Service if the connection string of the specified data store were to be set to the string being tested, together with extra diagnostics information for the specified connection string.

**DBRejectedConnection**

The database rejected the logon attempt from the Monitor Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**

The expected stored procedures are missing from the database. This may be because the Monitor Service schema has not been added to the database.

**DBNotFound**

The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**

The version of the Monitor Service currently in use is incompatible with the version of the Monitor Service schema on the database. Upgrade the Monitor Service to a more recent version.

**DBOlderVersionThanService**

The version of the Monitor Service schema on the database is incompatible with the version of the Monitor Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**

A database schema upgrade is currently in progress.

**OK**

The Set-MonitorDBConnection command would succeed if it were executed with the supplied connection string.

**Failed**

The Monitor Service has failed.

**Unknown**

The status of the Monitor Service cannot be determined.

### Notes

If the command fails, the following errors can be returned.

### Error Codes

**InvalidDBConnectionString**

The database connection string has an invalid format.
DatabaseError
An error occurred in the service while attempting a database operation.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Test-MonitorDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Tests a database connection string for the Monitor Service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Test-MonitorDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Tests an invalid database connection string for the Monitor Service.
Citrix.Storefront.Admin.V1

Oct 10, 2013

Overview

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SfStorefrontSnapin</td>
<td>This service short description</td>
</tr>
<tr>
<td>Sf Filtering</td>
<td>Describes the common filtering options for XenDesktop cmdlets.</td>
</tr>
</tbody>
</table>

Cmdlets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-SfServerToCluster</td>
<td>Adds a new server to an existing cluster.</td>
</tr>
<tr>
<td>Add-SfStorefrontAddress</td>
<td>Modifies a StoreFront address configuration by adding an additional StoreFront address to it.</td>
</tr>
<tr>
<td>Get-SfCluster</td>
<td>Gets all Storefront clusters present in the site.</td>
</tr>
<tr>
<td>Get-SfDBConnection</td>
<td>Gets the database string for the specified data store used by the Storefront Service.</td>
</tr>
<tr>
<td>Get-SfDBSchema</td>
<td>Gets a script that creates the Storefront Service database schema for the specified data store.</td>
</tr>
<tr>
<td>Get-SfDBVersionChangeScript</td>
<td>Gets a script that updates the Storefront Service database schema.</td>
</tr>
<tr>
<td>Get-SfInstalledDBVersion</td>
<td>Gets a list of all available database schema versions for the Storefront Service.</td>
</tr>
<tr>
<td>Get-SfIsStorefrontInstalled</td>
<td>Tells whether StoreFront Services and Privileged Service are installed.</td>
</tr>
<tr>
<td>Get-SfService</td>
<td>Gets the service record entries for the Storefront Service.</td>
</tr>
<tr>
<td>Get-SfServiceAddedCapability</td>
<td>Gets any added capabilities for the Storefront Service on the controller.</td>
</tr>
<tr>
<td>Get-SfServiceInstance</td>
<td>Gets the service instance entries for the Storefront Service.</td>
</tr>
<tr>
<td>Get-SfServiceStatus</td>
<td>Gets the current status of the Storefront Service on the controller.</td>
</tr>
<tr>
<td>Get-SfStorefrontAddress</td>
<td>Gets the high-level description of a configuration for StoreFront addresses, based on a configuration byte array.</td>
</tr>
<tr>
<td>Get-SfTask</td>
<td>Gets the task history for the Storefront Service.</td>
</tr>
<tr>
<td>New-SfCluster</td>
<td>Creates new Storefront cluster with default set of services.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>New-SfStorefrontAddress</td>
<td>Creates a new StoreFront address configuration, specifying a single address.</td>
</tr>
<tr>
<td>Remove-SfServerFromCluster</td>
<td>Removes server from the cluster.</td>
</tr>
<tr>
<td>Remove-SfServiceMetadata</td>
<td>Removes metadata from the given service.</td>
</tr>
<tr>
<td>Remove-SfTask</td>
<td>Removes from the database completed tasks for the Storefront Service.</td>
</tr>
<tr>
<td>Remove-SfTaskMetadata</td>
<td>Removes metadata from the given Task.</td>
</tr>
<tr>
<td>Reset-SfServiceGroupMembership</td>
<td>Reloads the access permissions and configuration service locations for the Storefront Service.</td>
</tr>
<tr>
<td>Set-SfCluster</td>
<td>Sets the parameters on the given cluster.</td>
</tr>
<tr>
<td>Set-SfDBConnection</td>
<td>Configures a database connection for the Storefront Service.</td>
</tr>
<tr>
<td>Set-SfServiceMetadata</td>
<td>Adds or updates metadata on the given service.</td>
</tr>
<tr>
<td>Set-SfTaskMetadata</td>
<td>Adds or updates metadata on the given Task.</td>
</tr>
<tr>
<td>Test-SfDBConnection</td>
<td>Tests a database connection for the Storefront Service.</td>
</tr>
</tbody>
</table>
about_SfStorefrontSnapin

Oct 10, 2013

TOPIC

about_SfStorefrontSnapin

SHORT DESCRIPTION

This service short description

COMMAND PREFIX

All commands in this snap-in have the noun prefixed with ‘Sf’.

LONG DESCRIPTION

This service long description
**about_Sf_Filtering**

Oct 10, 2013

**TOPIC**

XenDesktop - Advanced Dataset Filtering

**SHORT DESCRIPTION**

Describes the common filtering options for XenDesktop cmdlets.

**LONG DESCRIPTION**

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

```
Get-<Noun> | Where { $_.Size = 'Small' } | Sort 'Date' | Select -First 10
```

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

```
Get-<Noun> -Size 'Small' -SortBy 'Date' -MaxRecordCount 10
```

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

- **-MaxRecordCount <int>**

  Specifies the maximum number of results to return.
  For example, to return only the first nine results use:

  ```powershell
  Get-<Noun> -MaxRecordCount 9
  ```

  If not specified, only the first 250 records are returned, and if more are available, a warning is produced:
WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

To retrieve all records, specify a large number for -MaxRecordCount. As the value is an integer, you can use the following:

```powershell
Get-<Noun> -MaxRecordCount [int]:MaxValue
```

- `ReturnTotalRecordCount [<SwitchParameter>]`

  When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

  ```powershell
  Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
  ....
  Get-<Noun>: Returned 9 of 10 items
  At line:1 char:18
  + Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
  + CategoryInfo          : OperationStopped: (:) [Get-<Noun>], PartialDataException
  + FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
  
  The count can be accessed using the TotalAvailableResultCount property:

  ```powershell
  $count = $error[0].TotalAvailableResultCount
  ```

- `Skip <int>`

  Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

- `SortBy <string>`

  Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.
Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-sortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values. Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

-sortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter 'Name -like "High**" -or (Priority -eq 1 -and Severity -ge 2)'

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

EXAMPLES

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company "citrix" -Product [X]EN**
Get-<Noun> -Product "Xen"** -Company "CITRIX"
Get-<Noun> -Filter { Company -eq 'Citrix' -and Product -like 'Xen'**}
See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
Get-<Noun> -Company "Abc['"]"                 # Matches Abc*
Get-<Noun> -Company "Abc"**                    # Matches Abc*
Get-<Noun> -Filter { Company -eq "Abc"** }     # Matches Abc*
Get-<Noun> -Filter { Company -eq "A""B"'C" }   # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

```
Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```
Get-<Noun> -Filter 'Enabled'      # Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled' # Equivalent to 'Enabled -eq $false'
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```
Get-<Noun> -Shape [Shapes]:Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle" }
Get-<Noun> -Filter { Shape -like 'C' }

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

```powershell
Get-<Noun> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z" }
$d = [DateTime]"2010-08-23T12:30:00.0Z"
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }
```

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

```powershell
Get-<Noun> -Filter { StartTime -ge '-2' } # Two days ago
Get-<Noun> -Filter { StartTime -ge '-1:30' } # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge '-0:0:30' } # 30 seconds ago
```

### ARRAY PROPERTIES

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings. Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

```powershell
Get-<Noun> -User Fred
Get-<Noun> -Filter { User -like "Fred" }
Get-<Noun> -Filter { Users -contains "Fred" }
```

You can also use the singular form with -Filter to search using other operators:

```powershell
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter { User -eq "Frederick" }
# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter { User -lt 'F' }
```

### COMPLEX EXPRESSIONS
When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

```
Get-<Noun> -Filter { Shape -eq 'Circle' -or Shape -eq 'Square' }
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }
```

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

```
Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue' -and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue' -and Shape -eq 'Circle') }
```

### PAGING

The simplest way to page through data is to use the -Skip and -MaxRecordCount parameters. So, to read the first three pages of data with 10 records per page, use:

```
Get-<Noun> -Skip 0 -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 10 -MaxRecordCount 10 <other filtering criteria>
Get-<Noun> -Skip 20 -MaxRecordCount 10 <other filtering criteria>
```

You must include the same filtering criteria on each call, and ensure that the data is sorted consistently.

The above approach is often acceptable, but as each call performs an independent query, data changes can result in records being skipped or appearing twice. One approach to improve this is to sort by a unique id field and then start the search for the next page at the unique id after the last unique id of the previous page. For example:

```
# Get the first page
Get-<Noun> -MaxRecordCount 10 -SortBy SerialNumber

SerialNumber ...  
------------- 
A120004 
A120007 
... 7 other records ... 
A120900 

# Get the next page
Get-<Noun> -MaxRecordCount 10 -Filter { FirstName -gt 'A120900' }

SerialNumber ... 
```
FILTER SYNTAX DEFINITION

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= "{" <ComponentList> "}"

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

    <Component>

<Component> ::= <NotOperator> <Factor> |

    <Factor>

<Factor> ::= "(" <ComponentList> ")" |

    <PropertyName> <ComparisonOperator> <Value> |
    <PropertyName>

<AndOrOperator> ::= "-and" | "-or"

<NotOperator> ::= "-not" | "!

<ComparisonOperator>

    ::= "-eq" | "-ne" | "-le" | "-ge" | "-lt" | "-gt" | "-like" | "-notlike" | "-contains" | "-notcontains" | "-in" | "-notin"

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

    <scalar variable> | <array variable> | "$null" | "$true" | "$false"
Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30" means 1 hour and 30 minutes ago.
Add-SfServerToCluster

Oct 10, 2013
Add a new server to an existing cluster.

Syntax

Add-SfServerToCluster -ClusterId <Guid> -ServerName <String> [-StorefrontUrl <Uri>] [-FarmName <String>] [-XmlServices <Uri[]>] [-RunAsynchronously <Boolean>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Adds a new server to an existing cluster. Optionally updates Farm and Storefront Url. After operation succeeds, all servers are configured identically.

Related topics

Get-SfCluster
New-SfCluster
Remove-SfServerFromCluster
Set-SfCluster

Parameters

-ClusterId <Guid>
The id of the cluster to perform operation on.

| Required? | true |
| Default Value |
| Accept Pipeline Input? | false |

-ServerName <String>
The name of the server to join to existing cluster. The name must be one of the values returned by Get-SfCluster

| Required? | true |
| Default Value |
| Accept Pipeline Input? | false |

-StorefrontUrl <Uri>
The url that will be used by Receivers to contact Storefront. Http or https absolute urls are accepted.
<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Server name and http binding.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-FarmName <String>**

Name of the farm that will be used within Store service. Either both FarmName and XmlServices need to be specified or none of them.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-XmlServices <Uri[]>**

Collection of the url of xml services that will be used inside a farm. The urls need to be http or https, be absolute and share the same schema and port. Either both FarmName and XmlServices need to be specified or none of them.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-RunAsynchronously <Boolean>**

If set, the command will run asynchronously.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-LoggingId <Guid>**

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**-AdminAddress**: `<String>`

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


Returns cluster description or a task, if ran asynchronously.

Examples

-------------------------- EXAMPLE 1 --------------------------

Add-SfServerToCluster -ClusterId (Guid) -ServerName NewSfServer -RunAsynchronously $true

Adds "NewSfServer" to cluster with id (Guid).
Add-SfStorefrontAddress

Oct 10, 2013

Modifies a StoreFront address configuration by adding an additional StoreFront address to it.

Syntax

Add-SfStorefrontAddress [-ByteArray] <Byte[]> -Name <String> -Url <String> -Enabled <Boolean> -Description <String> [<CommonParameters>]

Detailed Description

Use this command to transform an existing StoreFront configuration into a new configuration, where the new configuration contains one additional address. The original configuration is supplied as an input, along with the properties of the new StoreFront address being added. The cmdlet outputs the modified configuration, which can then be passed to the Citrix Broker Service using the Add-BrokerMachineConfiguration command.

This command does not, by itself, have any persistent effects within XenDesktop. To make the change persistent, the new configuration byte array must first be transformed into a machine configuration within the Citrix Broker Service. To do this, use the New-BrokerMachineConfiguration command. You can then use the Add-BrokerMachineConfiguration and Set-BrokerMachineConfiguration commands to fully associate the new configuration with a delivery group.

Related topics

New-SfStorefrontAddress
Get-SfStorefrontAddress
New-BrokerMachineConfiguration
Add-BrokerMachineConfiguration
Set-BrokerMachineConfiguration

Parameters

[ByteArray]<Byte[]>

Specifies the initial configuration, on which the new configuration is based. All of the addresses in the original configuration are also present in the new configuration, along with the additional address specified. This configuration byte array is obtained from an earlier call to New-SfStorefrontAddress, or from Get-BrokerMachineConfiguration.

Required? true
Default Value
Accept Pipeline Input? true [ByValue]

-Name<String>

Specifies the name of the new StoreFront.

Required? true
Default Value
Accept Pipeline Input? false

-Url<String>

Specifies the URL to the StoreFront, such as "https://mysite.com/Citrix/StoreWeb".

Required? true
Default Value
Accept Pipeline Input? false

-Enabled<Boolean>

Specifies if the new StoreFront address should be enabled for user access.

Required? true
Default Value
Accept Pipeline Input? false

-Description<String>

Specifies a human-readable description of the new StoreFront.

Required? true
Default Value
Accept Pipeline Input?  false

Input Type
System.Byte[] This cmdlet accepts configurations as pipeline input, as an alternative to supplying the ByteArray parameter.

Return Values
System.Byte[]
This cmdlet outputs the new, modified configuration. This differs from the original configuration in that it contains the additional StoreFront address.

Examples
-------------------------- EXAMPLE 1 --------------------------
C:\PS> $newConfiguration = Add-SfStorefrontAddress -ByteArray $originalConfiguration -Url "https://mysite.com/Citrix/StoreWeb" -Description "This StoreFront delivers my corporate applications" -Name "StoreFront1" -Enabled $true
This command transforms the configuration byte array specified by $originalConfiguration, adds the new StoreFront details, and stores the resulting configuration in $newConfiguration.
Get-SfCluster

Oct 10, 2013
Gets all Storefront clusters present in the site.

Syntax
Get-SfCluster [-AdminAddress <String>] [CommonParameters]

Detailed Description
Gets all Storefront clusters present in the site. There is one special Cluster with null id that lists the servers that are not part to any cluster (they are available to join any).

Related topics
New-SfCluster
Add-SfServerToCluster
Remove-SfServerFromCluster
Set-SfCluster

Parameters

AdminAddress

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>localhost</td>
<td>false</td>
</tr>
</tbody>
</table>

Required? | false

Default Value | localhost. Once a value is provided by any cmdlet, this value becomes the default.

Return Values
Citrix.Storefront.DataModel.Cluster[

Returns array of clusters available in current deployment.
Get-SfDBConnection

Oct 10, 2013

Gets the database string for the specified data store used by the Storefront Service.

Syntax

Get-SfDBConnection [-AdminAddress <String>] [CommonParameters]

Detailed Description

Returns the database connection string for the specified data store.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

Related topics

Get-SfServiceStatus

Set-SfDBConnection

Test-SfDBConnection

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? false

Return Values

system.string

The database connection string configured for the Storefront Service.

Notes

If the command fails, the following errors can be returned.

Error Codes

--------

NoDBConnections
The database connection string for the Storefront Service has not been specified.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

------------------------ EXAMPLE 1 ------------------------

c:\PS>Get-SfDBConnection

Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True
Get the database connection string for the Storefront Service.
Get-SfDBSchema

Oct 10, 2013

Gets a script that creates the Storefront Service database schema for the specified data store.

Syntax

Get-SfDBSchema [-DatabaseName <String>] [-ServiceGroupName <String>] [-ScriptType <ScriptTypes>] [-LocalDatabase] [-Sid <String>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Gets SQL scripts that can be used to create a new Storefront Service database schema, add a new Storefront Service to an existing site, or create a database server logon for a Storefront Service. If no Sid parameter is provided, the scripts obtained relate to the currently selected Storefront Service instance, otherwise the scripts relate to Storefront Service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied. The current service instance is that on the local machine, or that explicitly specified by the last usage of the -AdminAddress parameter to a Storefront SDK cmdlet. The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured. The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SMDCMD mode'. The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to Storefront Service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to Storefront Service roles

If ScriptType is Evict, the returned script contains:

- Removal of Storefront Service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

If the service uses two data stores they can exist in the same database. You do not need to configure a database before using this command.
Related topics

Set-SfDBConnection

Parameters

- **DatabaseName**:<String>

  Specifies the name of the database for which the schema will be generated.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

- **ServiceGroupName**:<String>

  Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the Storefront services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

- **ScriptType**:<ScriptTypes>

  Specifies the type of database script returned. Available script types are:

  Database

  Returns a full database script that can be used to create a database schema for the Storefront Service in a database instance that does not already contain a schema for this service. The DatabaseName and ServiceGroupName parameters must be specified to create a script of this type.

  Instance

  Returns a permissions script that can be used to add further Storefront services to an existing database instance that already contains the full Storefront service schema, associating the services to the Service Group. The Sid parameter can optionally be specified to create a script of this type.

  Login

  Returns a database logon script that can be used to add the required logon accounts to an existing database instance that contains the Storefront Service schema. This is used primarily when creating a mirrored database environment. The DatabaseName parameter must be specified to create a script of this type.

  Evict
Returns a script that can be used to remove the specified Storefront Service from the database entirely. The DatabaseName and Sid parameters must be specified to create a script of this type.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>DatabaseName</td>
<td>false</td>
<td>Database</td>
<td>false</td>
</tr>
<tr>
<td>Sid</td>
<td>false</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **LocalDatabase**<SwitchParameter>

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for Storefront services.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **Sid**<String>

Specifies the SID of the controller on which the Storefront Service instance to remove from the database is running.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Return Values

System.string

A string containing the required SQL script for application to a database.

Notes
The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

Create the database schema.
Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

Create the user and the role (providing the schema does not already exist).
Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned.

Error Codes
-----------
GetSchemasFailed
The database schema could not be found.
ActiveDirectoryAccountResolutionFailed
The specified Active Directory account or Group could not be found.
DatabaseError
An error occurred in the service while attempting a database operation.
DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.
DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.
PermissionDenied
You do not have permission to execute this command.
AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

```powershell
Get-SfDBSchema -DatabaseName MyDB -ServiceGroupName MyServiceGroup > c:\SfSchema.sql
```
Get the full database schema for site data store of the Storefront Service and copy it to a file called 'c:\SfSchema.sql'.

This script can then be used to create the schema in a pre-existing database named 'MyDB' that does not already contain a Storefront Service site schema.

-------------------------- EXAMPLE 2 --------------------------

```powershell
Get-SfDBSchema -DatabaseName MyDB -scriptType Login > c:\StorefrontLogins.sql
```
Get the logon scripts for the Storefront Service.
Get-SfDBVersionChangeScript

Oct 10, 2013

_gets a script that updates the Storefront Service database schema._

Syntax

Get-SfDBVersionChangeScript -DatabaseName <String> -TargetVersion <Version> [-AdminAddress <String>] [-<CommonParameters>]

Detailed Description

_Returns a database script that can be used to upgrade or downgrade the site or secondary schema for the Storefront Service from the current schema version to a different version._

Related topics

Get-SfInstalledDBVersion

Parameters

- **-DatabaseName** <String>

Specifies the name of the database instance to which the update applies.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-TargetVersion** <Version>

Specifies the version of the database you want to update to.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress** <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values

System.Management.Automation.PSObject

A PSObject containing the required SQL script for application to a database.

Notes

The PSObject returned by this cmdlet contains the following properties:

-- Script The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

-- NeedExclusiveAccess Indicates whether all services in the service group must be shut down during the update or not.

-- CanUndo Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any Storefront services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-SfServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned.

Error Codes

----------

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the Storefront Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $update = Get-SfDBVersionChangeScript -DatabaseName MyDb -TargetVersion 1.0.75.0
C:\PS> $update.Script > update_75.sql

Gets an SQL update script to update the current schema to version 1.0.75.0. The resulting update_75.sql script is suitable for direct use with the SQL Server SQLCMD utility.
Get-SfInstalledDBVersion

Oct 10, 2013
Gets a list of all available database schema versions for the Storefront Service.

Syntax

Get-SfInstalledDBVersion [-Upgrade] [-Downgrade] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Returns the current version of the Storefront Service database schema, if no flags are set, otherwise returns versions for which upgrade or downgrade scripts are available and have been stored in the database.

Related topics

Parameters

- **Upgrade**<SwitchParameter>

  Specifies that only schema versions to which the current database version can be updated should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **Downgrade**<SwitchParameter>

  Specifies that only schema versions to which the current database version can be reverted should be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**<String>

  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values
System\Version

The Get-SfInstalledDbVersion command returns objects containing the new definition of the Storefront Service database schema version.

Major <Integer>
Minor <Integer>
Build <Integer>
Revision <Integer>

Notes
If the command fails, the following errors can be returned.

Error Codes
-----------

InvalidParameterCombination
Both the Upgrade and Downgrade flags were specified.

NoOp
The operation was successful but had no effect.

NoDBConnections
The database connection string for the Storefront Service has not been specified.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-SfInstalledDBVersion

Major Minor Build Revision
---- ---- ---- -------
 5  6  0  0
Get the currently installed version of the Storefront Service database schema.

-------------------------- EXAMPLE 2 --------------------------

c:\PS>Get-SfInstalledDBVersion -Upgrade

Major Minor Build Revision
---- ---- ---- -------
 6  0  0  0
Get the versions of the Storefront Service database schema for which upgrade scripts are supplied.
Get-SfIsStorefrontInstalled

Oct 10, 2013
Tells whether StoreFront Services and Privileged Service are installed.

Syntax

Get-SfIsStorefrontInstalled [-AdminAddress <String>] [CommonParameters]

Detailed Description

Related topics

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values

bool

True if both StoreFront Services and Privileged Service are installed, false otherwise.
Get-SfService

Oct 10, 2013
Gets the service record entries for the Storefront Service.

Syntax


Detailed Description

Returns instances of the Storefront Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Related topics

Parameters

-Property <String[]>

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

| Required? | false |
| Default Value | |
| Accept Pipeline Input? | false |

-ReturnTotalRecordCount <SwitchParameter>

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Sf_Filtering for details.

| Required? | false |
| Default Value | False |
| Accept Pipeline Input? | false |

-MaxRecordCount <Int32>

 Specifies the maximum number of records to return.

| Required? | false |
| Default Value | 250 |
-Skip<Int32>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |

-SortBy<String>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |

-Filter<String>

Gets records that match a PowerShell-style filter expression. See about_Sf_Filtering for details.

| Required? | false |
| Accept Pipeline Input? | false |

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Return Values

Citrix.Storefront.Sdk.Service
The `Get-SfServiceInstance` command returns an object containing the following properties.

**Uid <Integer>**
Specifies the unique identifier for the service in the group. The unique identifier is an index number.

**ServiceHostId <Guid>**
Specifies the unique identifier for the service instance.

**DNSName <String>**
Specifies the domain name of the host on which the service runs.

**MachineName <String>**
Specifies the short name of the host on which the service runs.

**CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>**
Specifies whether the service is running, started but inactive, stopped, or failed.

**LastStartTime <DateTime>**
Specifies the date and time at which the service was last restarted.

**LastActivityTime <DateTime>**
Specifies the date and time at which the service was last stopped or restarted.

**OSType**
Specifies the operating system installed on the host on which the service runs.

**OSVersion**
Specifies the version of the operating system installed on the host on which the service runs.

**ServiceVersion**
Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

**DatabaseUserName <string>**
Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

**Sid <string>**
Specifies the Active Directory account security identifier for the machine on which the service instance is running.

**ActiveSiteServices <string[]>**
Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.
Notes
If the command fails, the following errors can be returned.

Error Codes
-----------

UnknownObject
One of the specified objects was not found.

PartialData
Only a subset of the available data was returned.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase
The query required to get the database was not defined.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Get-SfService

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uid</td>
<td>1</td>
</tr>
<tr>
<td>ServiceHostId</td>
<td>ae6f464-f1ee-4042-a523-66982e0cecd0</td>
</tr>
<tr>
<td>DNSName</td>
<td>MyServer.company.com</td>
</tr>
<tr>
<td>MachineName</td>
<td>MYSERVER</td>
</tr>
<tr>
<td>CurrentState</td>
<td>On</td>
</tr>
<tr>
<td>LastStartTime</td>
<td>04/04/2011 15:25:38</td>
</tr>
<tr>
<td>LastActivityTime</td>
<td>04/04/2011 15:33:39</td>
</tr>
<tr>
<td>OSType</td>
<td>Win32NT</td>
</tr>
<tr>
<td>OSVersion</td>
<td>6.1.7600.0</td>
</tr>
<tr>
<td>ServiceVersion</td>
<td>5.1.0.0</td>
</tr>
<tr>
<td>DatabaseUserName</td>
<td>NT AUTHORITY\NETWORK SERVICE</td>
</tr>
<tr>
<td>SID</td>
<td>S-1-5-21-2316621082-1546847349-2782505528-1165</td>
</tr>
<tr>
<td>ActiveSiteServices</td>
<td>[MySiteService1, MySiteService2...]</td>
</tr>
</tbody>
</table>

Get all the instances of the Storefront Service running in the current service group.
Get-SfServiceAddedCapability

Oct 10, 2013

Gets any added capabilities for the Storefront Service on the controller.

Syntax

Get-SfServiceAddedCapability [-AdminAddress <String>] [CommonParameters]

Detailed Description

Enables updates to the Storefront Service on the controller to be detected.
You do not need to configure a database connection before using this command.

Related topics

Parameters

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Default Value

Localhost. Once a value is provided by any cmdlet, this value becomes the default.

Accept Pipeline Input?

false

Return Values

System.String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.2489
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-SfServiceAddedCapability
Get the added capabilities of the Storefront Service.
Get-SfServiceInstance

Oct 10, 2013
Gets the service instance entries for the Storefront Service.

Syntax
Get-SfServiceInstance [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Returns service interfaces published by the instance of the Storefront Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Related topics
Parameters
-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Accept Pipeline Input? | false

Return Values
Citrix.Storefront.Sdk.ServiceInstance

The Get-SfServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.
ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Sf.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations
InterService - for operations between different services
Peer - for communications between services of the same type

Metadata <Citrix.Storefront.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

---------

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-SfServiceInstance

Address            : http://MyServer.com:80/Citrix/StorefrontContract
Binding            : wcf_HTTP_kerb
InterfaceType      : SDK
Metadata           :
MetadataMap        :
ServiceAccount     : ENG\MyAccount$
ServiceAccountSid  : S-1-5-21-2406005612-3133289213-1653143164
ServiceGroupName   : MyServiceGroup
ServiceGroupUid    : a88d2f6b-c00a-4f76-9c38-e8330093b54d
ServiceInstanceUid : 00000000-0000-0000-0000-000000000000
ServiceType        : Sf
Version            : 1

Address            : http://MyServer.com:80/Citrix/StorefrontContract/IServiceApi
Binding            : wcf_HTTP_kerb
InterfaceType      : InterService
Metadata           :
MetadataMap        :
ServiceAccount     : ENG\MyAccount
ServiceAccountSid  : S-1-5-21-2406005612-3133289213-1653143164
Get all instances of the Storefront Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.
Get-SfServiceStatus

Oct 10, 2013
Gets the current status of the Storefront Service on the controller.

Syntax

Get-SfServiceStatus [-AdminAddress <String>] [CommonParameters]

Detailed Description

Enables the status of the Storefront Service on the controller to be monitored. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured.

Related topics

Set-SfDBConnection
Test-SfDBConnection
Get-SfDBConnection
Get-SfDBSchema

Parameters

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


The Get-SfServiceStatus command returns an object containing the status of the Storefront Service together with extra diagnostics information.

DBUnconfigured

The Storefront Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the Storefront Service. This may be because the service attempted to log
on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured
The expected stored procedures are missing from the database. This may be because the Storefront Service schema has not been added to the database.

DBNotFound
The specified database could not be located with the configured connection string.

DBNewerVersionThanService
The version of the Storefront Service currently in use is incompatible with the version of the Storefront Service schema on the database. Upgrade the Storefront Service to a more recent version.

DBOlderVersionThanService
The version of the Storefront Service schema on the database is incompatible with the version of the Storefront Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress
A database schema upgrade is currently in progress.

OK
The Storefront Service is running and is connected to a database containing a valid schema.

Failed
The Storefront Service has failed.

Unknown
(0) The service status cannot be determined.

Notes
If the command fails, the following errors can be returned.

Error Codes

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for
various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

------------------------------- EXAMPLE 1 -------------------------------

c:PS>Get-SfServiceStatus

DBUnconfigured
Get the current status of the Storefront Service.
Get-SfStorefrontAddress

Oct 10, 2013

Gets the high-level description of a configuration for StoreFront addresses, based on a configuration byte array.

**Syntax**

```
Get-SfStorefrontAddress [-ByteArray] <Byte[]> [<CommonParameters>]
```

**Detailed Description**

Use this command to convert a configuration byte array into a set of named property settings. The byte array will either have been retrieved from the Citrix Broker Service, or from the New-SfStorefrontAddress cmdlet.

**Related topics**

- New-SfStorefrontAddress
- Add-SfStorefrontAddress
- New-BrokerMachineConfiguration
- Add-BrokerMachineConfiguration
- Set-BrokerMachineConfiguration

**Parameters**

- **-ByteArray**<Byte[]>

  Specifies the low-level byte array (blob) to be interpreted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**Input Type**

System.Byte[]. The cmdlet accepts the ByteArray parameter as pipeline input.

**Return Values**

Citrix.Storefront.Sdk.SfStorefrontAddress

This cmdlet outputs one SfStorefrontAddress object for each address that is configured within the slot. Each object has the following properties:

- **Name** - Specifies the name of the StoreFront.
- **Url** - Specifies the URL to the StoreFront, such as "https://mysite.com/Citrix/StoreWeb".
- **Enabled** - Specifies whether the StoreFront is enabled for users.
- **Description** - Specifies the human-readable name of the StoreFront.

**Examples**

```
----------- EXAMPLE 1 -----------
C:\PS> $configuration = New-SfStorefrontAddress -Url "https://mysite.com/Citrix/StoreWeb" -Description "This StoreFront delivers my corporate applications" -Name "StoreFront1" -Enabled $true
C:\PS> Get-SfStorefrontAddress -ByteArray $configuration

Name               | Url                                       | Enabled | Description                                      |
--------------------|-------------------------------------------|---------|--------------------------------------------------|
StoreFront1         | https://mysite.com/Citrix/StoreWeb        | True    | This StoreFront delivers my corporate applications.
```

This example shows a new configuration byte array being created to specify a single StoreFront address. The configuration byte array is then provided as input to the Get-SfStorefrontAddress command, which interprets and outputs the same fields.
Get-SfTask

Oct 10, 2013
Gets the task history for the Storefront Service.

Syntax

Detailed Description
Returns a list of tasks that have run or are currently running within the Storefront Service.

Related topics
Remove-SfTask
Add-SfTaskMetadata
Remove-SfTaskMetadata

Parameters
-TaskId<Guid>
Specifies the task identifier to be returned.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-ReturnTotalRecordCount<SwitchParameter>
When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Sf_Filtering for details.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>False</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-MaxRecordCount<Int32>
Specifies the maximum number of records to return.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>250</td>
</tr>
<tr>
<td>---------------</td>
<td>-----</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Skip<Int32>**

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

| Required? | false |
| Default Value | 0 |
| Accept Pipeline Input? | false |

**-SortBy<String>**

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

| Required? | false |
| Default Value | The default sort order is by name or unique identifier. |
| Accept Pipeline Input? | false |

**-Filter<String>**

Gets records that match a PowerShell-style filter expression. See about_Sf_Filtering for details.

| Required? | false |
| Accept Pipeline Input? | false |

**-AdminAddress<String>**

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

| Required? | false |
| Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default. |
| Accept Pipeline Input? | false |

Notes
If the command fails, the following errors can be returned.

Error Codes
----------

UnknownObject
One of the specified objects was not found.

PartialData
Only a subset of the available data was returned.

InvalidFilter
A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase
The query required to get the database was not defined.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration service.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.
New-SfCluster

Oct 10, 2013
Creates new Storefront cluster with default set of services.

Syntax

New-SfCluster -ServerName <String> -FarmName <String> -XmlServices <Uri[]> [-StorefrontUrl <Uri>] [-RunAsynchronously <Boolean>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

The server will have a default set of services containing fully-functionally Storefront server with Authentication, Store, Receiver for Web and Desktop Appliance.

Related topics
Get-SfCluster
Add-SfServerToCluster
Remove-SfServerFromCluster
Set-SfCluster

Parameters

-ServerName <String>

The name of the server to build a cluster on. The name must be one of the values returned by Get-SfCluster

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-FarmName <String>

Name of the farm that will be used within Store service.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

-XmlServices <Uri[]>

Collection of the url of xml services that will be used inside a farm. The urls neeed to be http or https, be absolute and share the same schema and port.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------</td>
</tr>
<tr>
<td><code>-StorefrontUrl</code></td>
<td><code>Uri</code></td>
</tr>
<tr>
<td><code>-RunAsynchronously</code></td>
<td><code>Boolean</code></td>
</tr>
<tr>
<td><code>-LoggingId</code></td>
<td><code>Guid</code></td>
</tr>
<tr>
<td><code>-AdminAddress</code></td>
<td><code>String</code></td>
</tr>
</tbody>
</table>

Return Values

Returns cluster description or a task, if ran asynchronously.

Examples

-------------------------- EXAMPLE 1 --------------------------

New-SfCluster -FarmName "XdSiteName" -XmlServices http://farm1,http://farm2 -ServerName SfServer -RunAsynchronously $true
Creates a new cluster on server "SfServer". The server will have a default set of services containing fully-functional Storefront server with Authentication, Store, Receiver for Web, Desktop Appliance site and the required infrastructure. The Store service will have a farm named "XdSiteName" that will contain servers farm1 and farm2, whose will be contacted using http on default port 80.
New-SfStorefrontAddress

Oct 10, 2013
Creates a new StoreFront address configuration, specifying a single address.

Syntax
New-SfStorefrontAddress -Name <String> -Url <String> -Enabled <Boolean> -Description <String> [-CommonParameters]

Detailed Description
Use this command when you want to create a new StoreFront configuration byte array from scratch, rather than modifying an existing one. You must define the URL for the StoreFront, and some additional details.

This command does not, by itself, have any persistent effects within XenDesktop. To make the change persistent, the new configuration byte array must first be transformed into a machine configuration within the Citrix Broker Service. To do this, use the New-BrokerMachineConfiguration command. You can then use the Add-BrokerMachineConfiguration and Set-BrokerMachineConfiguration commands to fully associate the new configuration with a delivery group.

Related topics
Add-SfStorefrontAddress
Get-SfStorefrontAddress
New-BrokerMachineConfiguration
Add-BrokerMachineConfiguration
Set-BrokerMachineConfiguration

Parameters
-Name <String>
Specifies the name of the new StoreFront.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-Url <String>
Specifies the URL to the StoreFront, such as "https://mysite.com/Citrix/StoreWeb".

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-Enabled <Boolean>
Specifies if the new StoreFront address should be enabled for user access.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-Description <String>
Specifies a human-readable description of the new StoreFront.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
The new configuration set, with all of the given modifications applied.

Examples

-------------------------- EXAMPLE 1 --------------------------

C:\PS> $configuration = New-SfStorefrontAddress -Url "https://mysite.com/Citrix/StoreWeb" -Description "This StoreFront delivers my corporate applications" -Name "StoreFront1" -Enabled $true

C:\PS> Get-SfStorefrontAddress -ByteArray $configuration

<table>
<thead>
<tr>
<th>Name</th>
<th>Url</th>
<th>Enabled</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StoreFront1</td>
<td><a href="https://mysite.com/Citrix/StoreWeb">https://mysite.com/Citrix/StoreWeb</a></td>
<td>True</td>
<td>This StoreFront delivers my corporate applications.</td>
</tr>
</tbody>
</table>

This example shows a new configuration byte array being created to specify a single StoreFront address. The configuration byte array is then provided as input to the Get-SfStorefrontAddress command, which interprets and outputs the same fields.
Remove-SfServerFromCluster

Oct 10, 2013
Removes server from the cluster.

Syntax

```powershell
Remove-SfServerFromCluster -ClusterId <Guid> -ServerName <String> [-StorefrontUrl <Uri>] [-FarmName <String>] [-XmlServices <Uri[]>] [-RunAsynchronously <Boolean>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
```

Detailed Description

Removes server from the cluster and propagates information to other servers. The configuration of the server is wiped out, so the server can be reused.

Related topics

- Get-SfCluster
- New-SfCluster
- Add-SfServerToCluster
- Set-SfCluster

Parameters

- `-ClusterId <Guid>`
  The id of the cluster to perform operation on.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

- `-ServerName <String>`
  The name of the server to remove from cluster. The name must be one of the values returned by Get-SfCluster.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

- `-StorefrontUrl <Uri>`
  The url that will be used by Receivers to contact Storefront. Http or https absolute urls are accepted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Required?</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------</td>
</tr>
<tr>
<td>FarmName</td>
<td>false</td>
</tr>
<tr>
<td>XmlServices</td>
<td>false</td>
</tr>
<tr>
<td>RunAsynchronously</td>
<td>false</td>
</tr>
<tr>
<td>LoggingId</td>
<td>false</td>
</tr>
</tbody>
</table>
**-AdminAddress**: String

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


Returns cluster description or a task, if ran asynchronously.

Examples

-------------------------- EXAMPLE 1 --------------------------

Remove-SfServerFromCluster -ClusterId (Guid) -ServerName BrokenSfServer -RunAsynchronously $true

Removes Server "BrokenSfServer" from a Cluster with id (Guid).
Remove-SfServiceMetadata

Oct 10, 2013
Removes metadata from the given service.

Syntax

Remove-SfServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-SfServiceMetadata [-ServiceHostId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-SfServiceMetadata [-InputObject] <Service[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Remove-SfServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability to remove metadata from the given service.

Related topics
Set-SfServiceMetadata

Parameters

-ServiceHostId <Guid>
Id of the service

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

-InputObject <Service[]>
Objects to which metadata is to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

-Map <PSObject>
Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
</tr>
</tbody>
</table>

**-Name**<String>

The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

Error Codes

-------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

------------------------------ EXAMPLE 1 ------------------------------

\c:\PS>Get-SfService | % { Remove-SfServiceMetadata -Map $_.MetadataMap }
Remove all metadata from all service objects.
Remove-SfTask

Oct 10, 2013
Removes from the database completed tasks for the Storefront Service.

Syntax

Remove-SfTask [-TaskId] <Guid> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Enables completed tasks that have run within the Storefront Service to be removed from the database.

Related topics

Get-SfTask
Add-SfTaskMetadata
Remove-SfTaskMetadata

Parameters

-TaskId<Guid>

Specifies the identifier for the task to be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>true (ByPropertyName)</td>
</tr>
</tbody>
</table>

-LoggingId<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default.
--- | ---
Accept Pipeline Input? | false

Input Type

System.Management.Automation.PSObject Objects containing the TaskId parameter can be piped to the Remove-SfTask command.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration service.

OperationDeniedByConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.
Remove-SfTaskMetadata

Oct 10, 2013
Removes metadata from the given Task.

Syntax

Remove-SfTaskMetadata [-TaskId] <Guid> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-SfTaskMetadata [-TaskId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-SfTaskMetadata [-InputObject] <Task[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Remove-SfTaskMetadata [-InputObject] <Task[]> -Name <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description
Provides the ability to remove metadata from the given Task.

Related topics
Set-SfTaskMetadata

Parameters

-TaskId<Guid>
Id of the Task

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue, ByPropertyName) |

-InputObject<Task[]>
Objects to which the metadata is to be added.

| Required? | true |
| Default Value | |
| Accept Pipeline Input? | true (ByValue) |

-Name<String>
The metadata property to remove.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-Map**<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

**-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Notes

If the command fails, the following errors can be returned.

Error Codes

----------
InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS>Get-SfTask | % { Remove-SfTaskMetadata -Map $_.MetadataMap }
Remove all metadata from all Task objects.
Reset-SfServiceGroupMembership

Oct 10, 2013
Reloads the access permissions and configuration service locations for the Storefront Service.

Syntax

Reset-SfServiceGroupMembership [-ConfigServiceInstance] <ServiceInstance[]> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Enables you to reload Storefront Service access permissions and configuration service locations. The Reset-SfServiceGroupMembership command must be run on at least one instance of the service type (Sf) after installation and registration with the configuration service. Without this operation, the Storefront services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-SfServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Related topics

Parameters

-ConfigServiceInstance <ServiceInstance[]>

Specifies the configuration service instance object that represents the service instance for the type 'InterService' that references a configuration service for the deployment.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true (ByValue)</td>
<td>false</td>
</tr>
</tbody>
</table>

-LoggingId <Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-AdminAddress <String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.2518
Input Type
Citrix.Storefront.Sdk.ServiceInstance[] Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-SfServiceGroupMembership command.

Notes
If the command fails, the following errors can be returned.

Error Codes
---------

NoSuitableServiceInstance
None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

--------------- EXAMPLE 1 ---------------
c:PS>Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-SfServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

--------------- EXAMPLE 2 ---------------
c:PS>Get-ConfigRegisteredServiceInstance -ServiceType Config -AdminAddress OtherServer.example.com | Reset-SfServiceGroupMembership
Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.
Set-SfCluster

Oct 10, 2013
Sets the parameters on the given cluster.

Syntax

Set-SfCluster -ClusterId <Guid> [-StorefrontUrl <Uri>] [-FarmName <String>] [-XmlServices <Uri>[-] [-RunAsynchronously <Boolean>] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Sets the parameters on the given cluster and propagate the changes to all servers within a given cluster.

Related topics

Get-SfCluster

New-SfCluster

Add-SfServerToCluster

Remove-SfServerFromCluster

Parameters

-ClusterId <Guid>

The id of the cluster to perform operation on.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-StorefrontUrl <Uri>

The url that will be used by Receivers to contact Storefront. Http or https absolute urls are accepted.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Server name and http binding.</td>
<td>false</td>
</tr>
</tbody>
</table>

-FarmName <String>

Name of the farm that will be used within Store service. Either both FarmName and XmlServices need to be specified or none of them.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>XmlServices</code>&lt;Uri[]&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Description</td>
<td>Collection of the url of xml services that will be used inside a farm. The urls need to be http or https, be absolute and share the same schema and port. Either both FarmName and XmlServices need to be specified or none of them.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>RunAsynchronously</code>&lt;Boolean&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Description</td>
<td>If set, the command will run asynchronously.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>LoggingId</code>&lt;Guid&gt;</td>
<td>false</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>Description</td>
<td>Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>AdminAddress</code>&lt;String&gt;</td>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
<tr>
<td>Description</td>
<td>Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Return Values


Returns cluster description or a task, if ran asynchronously.

Examples

-------------------------- EXAMPLE 1 --------------------------

Set-SfCluster -StoreFrontUrl http://SfUrl -ClusterId (Guid) -RunAsynchronously $true
Sets a Storefront Url in a Cluster with id (Guid). Propagates changes to all servers that are part of the cluster.
Set-SfDBConnection

Oct 10, 2013
Configures a database connection for the Storefront Service.

Syntax

Set-SfDBConnection [-DBConnection] <String> [-Force] [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Configures a connection to a database in which the Storefront Service can store its state. The service will attempt to connect and start using the database immediately after the connection is configured. The database connection string is updated to the specified value regardless of whether it is valid or not. Specifying an invalid connection string prevents a service from functioning until the error is corrected.

After a connection is configured, you cannot alter it without first clearing it (by setting the connection to $null).

You do not need to configure a database connection to use this command.

Related topics

Get-SfServiceStatus
Get-SfDBConnection
Test-SfDBConnection

Parameters

- **-DBConnection**<String>

  Specifies the database connection string to be used by the Storefront Service. Passing in $null will clear any existing database connection configured.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Force**<SwitchParameter>

  If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-LoggingId**<Guid>

  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>
**-AdminAddress[String]**

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

**Return Values**


The Set-SfDBConnection command returns an object containing the status of the Storefront Service together with extra diagnostics information.

**DBUnconfigured**
The Storefront Service does not have a database connection configured.

**DBRejectedConnection**
The database rejected the logon attempt from the Storefront Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

**InvalidDBConfigured**
The expected stored procedures are missing from the database. This may be because the Storefront Service schema has not been added to the database.

**DBNotFound**
The specified database could not be located with the configured connection string.

**DBNewerVersionThanService**
The version of the Storefront Service currently in use is incompatible with the version of the Storefront Service schema on the database. Upgrade the Storefront Service to a more recent version.

**DBOlderVersionThanService**
The version of the Storefront Service schema on the database is incompatible with the version of the Storefront Service currently in use. Upgrade the database schema to a more recent version.

**DBVersionChangeInProgress**
A database schema upgrade is currently in progress.

**OK**
The Storefront Service is running and is connected to a database containing a valid schema.

**Failed**
The Storefront Service has failed.

Unknown

The status of the Storefront Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

-----------

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured. After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS->Set-SfDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Configures a database connection string for the Storefront Service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS->Set-SfDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Configures an invalid database connection string for the Storefront Service.
Set-SfServiceMetadata

Oct 10, 2013
Adds or updates metadata on the given service.

Syntax

Set-SfServiceMetadata [-ServiceHostId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-SfServiceMetadata [-ServiceHostId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Set-SfServiceMetadata [-InputObject] <Service[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Allows you to store additional custom data against given service objects.

Related topics

Remove-SfServiceMetadata

Parameters

- **ServiceHostId** <Guid>
  
  Id of the service

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **InputObject** <Service[]>
  
  Objects to which metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name** <String>
Specifies the property name of the metadata to be added. The property must be unique for the service specified. The property cannot contain any of the following characters \/;#:.*?=<>|[]()’

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Value**<String>

Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-Map**<PSObject>

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-LoggingId**<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **-AdminAddress**<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

Return Values


Set-SfServiceMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>

Specifies the name of the property.

Value <string>

Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

------------

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.
ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------

c:\PS> Set-SfServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Set-SfTaskMetadata

Oct 10, 2013
Adds or updates metadata on the given Task.

Syntax

Set-SfTaskMetadata [-TaskId] <Guid> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-SfTaskMetadata [-TaskId] <Guid> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-SfTaskMetadata [-InputObject] <Task[]> -Map <PSObject> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]
Set-SfTaskMetadata [-InputObject] <Task[]> -Name <String> -Value <String> [-LoggingId <Guid>] [-AdminAddress <String>] [<CommonParameters>]

Detailed Description

Provides the ability for additional custom data to be stored against given Task objects.

Related topics

Remove-SfTaskMetadata

Parameters

- **-TaskId**<Guid>
  
  Id of the Task

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue, ByPropertyName)</td>
</tr>
</tbody>
</table>

- **-InputObject**<Task[]>
  
  Objects to which the metadata is to be added.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **-Map**<PSObject>
Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1" = "val1"; "name2" = "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>true (ByValue)</td>
</tr>
</tbody>
</table>

- **Name**:<String>

Specifies the property name of the metadata to be added. The property must be unique for the Task specified. The property cannot contain any of the following characters \/;#:.*?=<>|[]"'

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **Value**:<String>

Specifies the value for the property.

<table>
<thead>
<tr>
<th>Required?</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **LoggingId**:<Guid>

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td>false</td>
</tr>
</tbody>
</table>

- **AdminAddress**:<String>

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Accept Pipeline Input?</td>
<td></td>
</tr>
</tbody>
</table>
Default Value | Localhost. Once a value is provided by any cmdlet, this value becomes the default.
--- | ---
Accept Pipeline Input? | false

Return Values


Set-SfTaskMetadata returns a dictionary containing the new (name, value)-pairs.

Key <string>
Specifies the name of the property.

Value <string>
Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

----------

InvalidParameterCombination
The cmdlet parameters are inconsistent.

UnknownObject
One of the specified objects was not found.

DatabaseError
An error occurred in the service while attempting a database operation.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied
You do not have permission to execute this command.

AuthorizationError
There was a problem communicating with the Citrix Delegated Administration Service.
ConfigurationLoggingError
The operation could not be performed because of a configuration logging error.

CommunicationError
There was a problem communicating with the remote service.

ExceptionThrown
An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

--------------- EXAMPLE 1 -------------------

```
c:\PS> Set-SfTaskMetadata -TaskId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value
```

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>value</td>
</tr>
</tbody>
</table>

Add metadata with a name of 'property' and a value of 'value' to the Task with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.
Test-SfDBConnection

Oct 10, 2013
Tests a database connection for the Storefront Service.

Syntax

Test-SfDBConnection [-DBConnection] <String> [-LoggingId <Guid>] [-AdminAddress <String>] [CommonParameters>

Detailed Description

Tests a connection to the database in which the Storefront Service can store its state. The service will attempt to connect to the database without affecting the current connection to the database.

You do not have to clear the connection to use this command.

Related topics

Get-SfServiceStatus
Get-SfDBConnection
Set-SfDBConnection

Parameters

-**DBConnection**<String>
  Specifies the database connection string to be tested by the Storefront Service.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>

-**LoggingId**<Guid>
  Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

-**AdminAddress**<String>
  Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

<table>
<thead>
<tr>
<th>Required?</th>
<th>Default Value</th>
<th>Accept Pipeline Input?</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>Localhost. Once a value is provided by any cmdlet, this value becomes the default.</td>
<td>false</td>
</tr>
</tbody>
</table>
Return Values


The Test-SfDBConnection command returns an object containing the status of the Storefront Service if the connection string of the specified data store were to be set to the string being tested, together with extra diagnostics information for the specified connection string.

DBRejectedConnection

The database rejected the logon attempt from the Storefront Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the Storefront Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBNewerVersionThanService

The version of the Storefront Service currently in use is incompatible with the version of the Storefront Service schema on the database. Upgrade the Storefront Service to a more recent version.

DBOlderVersionThanService

The version of the Storefront Service schema on the database is incompatible with the version of the Storefront Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Set-SfDBConnection command would succeed if it were executed with the supplied connection string.

Failed

The Storefront Service has failed.

Unknown

The status of the Storefront Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

--------

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.
AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Examples

-------------------------- EXAMPLE 1 --------------------------
c:\PS>Test-SfDBConnection -DBConnection "Server=serverName\SQLEXPRESS;Initial Catalog = databaseName; Integrated Security = True"
OK
Tests a database connection string for the Storefront Service.

-------------------------- EXAMPLE 2 --------------------------
c:\PS>Test-SfDBConnection -DBConnection "Invalid Connection String"
Invalid database connection string format.
Tests an invalid database connection string for the Storefront Service.
Policies contain settings that are applied when the policy is enforced. You configure these settings using Studio in XenDesktop. If required, you can also use Active Directory Group Policy Editor.

The descriptions for each policy setting include the following information:
- The name of the policy setting
- The Citrix products to which the policy setting applies
- The additional settings, if applicable, required to enable a particular feature
- Other settings that are similar to the policy setting in question, if applicable

The following tables present settings you can configure within a policy. Find the task you want to perform in the left column, then locate its corresponding setting in the right column.

### Audio

<table>
<thead>
<tr>
<th>To perform this task:</th>
<th>Use this policy setting:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control whether to allow the use of multiple audio devices</td>
<td>Audio Plug N Play</td>
</tr>
<tr>
<td>Control whether to allow audio input from microphones on the user device</td>
<td>Client microphone redirection</td>
</tr>
<tr>
<td>Control audio quality on the user device</td>
<td>Audio quality</td>
</tr>
<tr>
<td>Control audio mapping to speakers on the user device</td>
<td>Client audio redirection</td>
</tr>
</tbody>
</table>

### Bandwidth for user devices

<table>
<thead>
<tr>
<th>To limit bandwidth used for:</th>
<th>Use this policy setting:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client audio mapping</td>
<td>● Audio redirection bandwidth limit or</td>
</tr>
<tr>
<td></td>
<td>● Audio redirection bandwidth limit percent</td>
</tr>
<tr>
<td>Cut-and-paste using local clipboard</td>
<td>● Clipboard redirection bandwidth limit or</td>
</tr>
<tr>
<td></td>
<td>● Clipboard redirection bandwidth limit percent</td>
</tr>
<tr>
<td>Access in a session to local client drives</td>
<td>● File redirection bandwidth limit or</td>
</tr>
<tr>
<td></td>
<td>● File redirection bandwidth limit percent</td>
</tr>
<tr>
<td>HDX MediaStream Multimedia Acceleration</td>
<td>● HDX MediaStream Multimedia Acceleration</td>
</tr>
<tr>
<td>To limit bandwidth used for:</td>
<td>Use this policy setting:</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td></td>
<td>HDX MediaStream Multimedia Acceleration bandwidth limit percent</td>
</tr>
<tr>
<td>Client session</td>
<td>Overall session bandwidth limit</td>
</tr>
<tr>
<td>Printing</td>
<td>Printer redirection bandwidth limit or Printer redirection bandwidth limit percent</td>
</tr>
<tr>
<td>TWAIN devices (such as a camera or scanner)</td>
<td>TWAIN device redirection bandwidth limit or TWAIN device redirection bandwidth limit percent</td>
</tr>
<tr>
<td>USB devices</td>
<td>Client USB device redirection bandwidth limit or Client USB device redirection bandwidth limit percent</td>
</tr>
</tbody>
</table>

Redirect of client drives and user devices

<table>
<thead>
<tr>
<th>To perform this task:</th>
<th>Use this policy setting:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control whether or not drives on the user device are connected when users log on to the server</td>
<td>Auto connect client drives</td>
</tr>
<tr>
<td>Control cut-and-paste data transfer between the server and the local clipboard</td>
<td>Client clipboard redirection</td>
</tr>
<tr>
<td>Control how drives map from the user device</td>
<td>Client drive redirection</td>
</tr>
<tr>
<td>Control whether users' local hard drives are available in a session</td>
<td>Client fixed drives and Client drive redirection</td>
</tr>
<tr>
<td>Control whether users' local floppy drives are available in a session</td>
<td>Client floppy drives and Client drive redirection</td>
</tr>
<tr>
<td>Control whether users' network drives are available in a session</td>
<td>Client network drives and Client drive redirection</td>
</tr>
<tr>
<td>Control whether users' local CD, DVD, or Blu-ray drives are available in a session</td>
<td>Client optical drives and Client drive redirection</td>
</tr>
<tr>
<td>Control whether users' local removable drives are available in</td>
<td>Client removable drives and</td>
</tr>
<tr>
<td>To perform this task</td>
<td>Use this policy setting</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Control whether users’ TWAIN devices, such as scanners and cameras, are available</td>
<td>● Client TWAIN device redirection</td>
</tr>
<tr>
<td>in a session and control compression of image data transfers</td>
<td>● TWAIN compression redirection</td>
</tr>
<tr>
<td>Control whether USB devices are available in a session</td>
<td>● Client USB device redirection and</td>
</tr>
<tr>
<td></td>
<td>● Client USB device redirection rules</td>
</tr>
<tr>
<td>Improve the speed of writing and copying files to a client disk over a WAN</td>
<td>Use asynchronous writes</td>
</tr>
</tbody>
</table>

### Content redirection

<table>
<thead>
<tr>
<th>To perform this task</th>
<th>Use this policy setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control whether to use content redirection from the server to the user device</td>
<td>Host to client redirection</td>
</tr>
</tbody>
</table>

### Desktop UI

<table>
<thead>
<tr>
<th>To perform this task</th>
<th>Use this policy setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control whether or not Desktop wallpaper is used in users' sessions</td>
<td>Desktop wallpaper</td>
</tr>
<tr>
<td>View window contents while a window is dragged</td>
<td>View window contents while dragging</td>
</tr>
</tbody>
</table>

### Graphics and multimedia

<table>
<thead>
<tr>
<th>To perform this task</th>
<th>Use this policy setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control the maximum number of frames per second sent to user devices from</td>
<td>Target frame rate</td>
</tr>
<tr>
<td>virtual desktops</td>
<td></td>
</tr>
<tr>
<td>Control the visual quality of images displayed on the user device</td>
<td>Visual quality</td>
</tr>
<tr>
<td>Control whether Flash content is rendered in sessions</td>
<td>Flash default behavior</td>
</tr>
</tbody>
</table>
Control whether websites can display Flash content when accessed in sessions | Use this policy setting:  
* Flash server-side content fetching URL list  
* Flash URL compatibility list

<table>
<thead>
<tr>
<th>Prioritize Multi-Stream network traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>To perform this task:</strong></td>
</tr>
<tr>
<td>Specify ports for ICA traffic across multiple connections and establish network priorities</td>
</tr>
<tr>
<td>Enable support for multi-stream connections between servers and user devices</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Print</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>To perform this task:</strong></td>
</tr>
<tr>
<td>Control creation of client printers on the user device</td>
</tr>
</tbody>
</table>
* Auto-create client printers and  
* Client printer redirection |
| Control the location where printer properties are stored | Printer properties retention |
| Control whether print requests are processed by the client or the server | Direct connections to print servers |
| Control whether users can access printers connected to their user devices | Client printer redirection |
| Control installation of native Windows drivers when automatically creating client and network printers | Automatic installation of in-box printer drivers |
| Control when to use the Universal Printer Driver | Universal print driver usage |
| Choose a printer based on a roaming user's session information | Default printer |
## Default policy settings

**Jan 06, 2014**  
**ICA**

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client clipboard redirection</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Desktop launches</td>
<td>Prohibited</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>ICA listener connection timeout</td>
<td>120000 milliseconds</td>
<td>XenDesktop 5, 5.5, 5.6 Feature Pack 1, 7 (Desktop OS), 7.1 (Desktop OS)</td>
</tr>
<tr>
<td>ICA listener port number</td>
<td>1494</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Launching of non-published programs during client connection</td>
<td>Prohibited</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>Read-only clipboard</td>
<td>Prohibited</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS)</td>
</tr>
</tbody>
</table>

### ICA/Audio

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio Plug N Play</td>
<td>Allowed</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>Audio quality</td>
<td>High - high definition audio</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Client audio redirection</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Client microphone redirection</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### ICA/Auto Client Reconnect

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Default setting</td>
<td>Applies to</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>----------------------------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>Auto client reconnect authentication</td>
<td>Do not require authentication</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Auto client reconnect logging</td>
<td>Do not log auto-reconnect events</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

**ICA/Bandwidth**

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio redirection bandwidth limit</td>
<td>0 Kbps</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Audio redirection bandwidth limit percent</td>
<td>0</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Client USB device redirection bandwidth limit</td>
<td>0 Kbps</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Client USB device redirection bandwidth limit percent</td>
<td>0</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Clipboard redirection bandwidth limit</td>
<td>0 Kbps</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Clipboard redirection bandwidth limit percent</td>
<td>0</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>COM port redirection bandwidth limit</td>
<td>0 Kbps</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>COM port redirection bandwidth limit percent</td>
<td>0</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>File redirection bandwidth limit</td>
<td>0 Kbps</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>File redirection bandwidth limit percent</td>
<td>0</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>HDX MediaStream Multimedia Acceleration bandwidth limit</td>
<td>0 Kbps</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
</tbody>
</table>

*Note: For XenDesktop 7.x, configure this setting using the registry.*
<table>
<thead>
<tr>
<th>Name</th>
<th>Default Setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDX MediaStream Multimedia Acceleration bandwidth limit percent</td>
<td>0 Kbps</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>LPT port redirection bandwidth limit</td>
<td>0 Kbps</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>LPT port redirection bandwidth limit percent</td>
<td>0</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Overall session bandwidth limit</td>
<td>0 Kbps</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Printer redirection bandwidth limit</td>
<td>0 Kbps</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Printer redirection bandwidth limit percent</td>
<td>0</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>TWAIN device redirection bandwidth limit</td>
<td>0 Kbps</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>TWAIN device redirection bandwidth limit percent</td>
<td>0</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
</tbody>
</table>

**ICA/Client Sensors**

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow applications to use the physical location of the client device</td>
<td>Prohibited</td>
<td>XenDesktop 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
</tbody>
</table>

**ICA/Desktop UI**

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desktop Composition Redirection</td>
<td>Enabled</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Desktop OS), 7.1 (Desktop OS)</td>
</tr>
<tr>
<td>Desktop Composition Redirection</td>
<td>Medium</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Desktop OS), 7.1</td>
</tr>
</tbody>
</table>
### Graphics Quality

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desktop wallpaper</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Menu animation</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>View window contents while dragging</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### ICA/End User Monitoring

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICA round trip calculation</td>
<td>Enabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>ICA round trip calculation interval</td>
<td>15 seconds</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>ICA round trip calculations for idle connections</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### ICA/Enhanced Desktop Experience

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enhanced Desktop Experience</td>
<td>Allowed</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
</tbody>
</table>

### ICA/File Redirection

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto connect client drives</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Client drive redirection</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Client fixed drives</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Client floppy drives</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Client network drives</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Name</td>
<td>Default setting</td>
<td>Applies to</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>Client optical drives</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Host to client redirection</td>
<td>Disabled</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>Preserve client drive letters</td>
<td>Disabled</td>
<td>XenDesktop 5, 5.5, 5.6 Feature Pack 1, 7 (Desktop OS), 7.1 (Desktop OS)</td>
</tr>
<tr>
<td>Read-only client drive access</td>
<td>Disabled</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Special folder redirection</td>
<td>Allowed</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Web Interface deployments only</td>
</tr>
<tr>
<td>Use asynchronous writes</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### ICA/Graphics

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display memory limit</td>
<td>65536 Kb</td>
<td>XenDesktop 5, 5.5, 5.6 Feature Pack 1, 7 (Desktop OS), 7.1 (Desktop OS)</td>
</tr>
<tr>
<td>Display mode degrade preference</td>
<td>Degrade color depth first</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Dynamic windows preview</td>
<td>Enabled</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Image caching</td>
<td>Enabled</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Legacy graphics mode</td>
<td>Disabled</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Name</td>
<td>Default setting</td>
<td>Applies to</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>Maximum allowed color depth</td>
<td>32 bits per pixel</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Notify user when display mode is degraded</td>
<td>Disabled</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>Queuing and tossing</td>
<td>Enabled</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

**ICA/Graphics/Caching**

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistent cache threshold</td>
<td>3000000 bps</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
</tbody>
</table>

**ICA/Keep Alive**

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICA keep alive timeout</td>
<td>60 seconds</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>ICA keep alives</td>
<td>Do not send ICA keep alive messages</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

**ICA/Local App Access**

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow local app access</td>
<td>Prohibited</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>URL redirection black list</td>
<td>No sites are specified</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>URL redirection white list</td>
<td>No sites are specified</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
</tbody>
</table>

**ICA/Mobile Experience**
<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic keyboard display</td>
<td>Prohibited</td>
<td>XenDesktop 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Launch touch-optimized desktop</td>
<td>Allowed</td>
<td>XenDesktop 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Remote the combo box</td>
<td>Prohibited</td>
<td>XenDesktop 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
</tbody>
</table>

### ICA/Multimedia

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limit video quality</td>
<td>Not configured</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Multimedia conferencing</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Optimization for Windows Media multimedia redirection over WAN</td>
<td>Allowed</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Use GPU for optimizing Windows Media multimedia redirection over WAN</td>
<td>Prohibited</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Windows Media client-side content fetching</td>
<td>Allowed</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Windows Media Redirection</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Windows Media Redirection buffer size</td>
<td>5 seconds</td>
<td>XenDesktop 5, 5.5, and 5.6, Feature Pack 1</td>
</tr>
<tr>
<td>Windows Media Redirection buffer size use</td>
<td>Disabled</td>
<td>XenDesktop 5, 5.5, and 5.6, Feature Pack 1</td>
</tr>
</tbody>
</table>

### ICA/Multi-Stream Connections
<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio over UDP</td>
<td>Allowed</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>Audio UDP port range</td>
<td>16500, 16509</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Multi-Port policy</td>
<td>Primary port (2598) has High Priority</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Multi-Stream computer setting</td>
<td>Disabled</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Multi-Stream user setting</td>
<td>Disabled</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
</tbody>
</table>

### ICA/Port Redirection

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto connect client COM ports</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Auto connect client LPT ports</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Client COM port redirection</td>
<td>Prohibited</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Client LPT port redirection</td>
<td>Prohibited</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

**Note:** For XenDesktop 7.x, configure this setting using the registry.

### ICA/Printing

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
</table>

---

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
### Client printer redirection

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default printer</td>
<td>Set default printer to the client’s main printer</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Printer assignments</td>
<td>User’s current printer is used as the default printer for the session</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Printer auto-creation event log preference</td>
<td>Log errors and warnings</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Session printers</td>
<td>No printers are specified</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Wait for printers to be created (desktop)</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

**ICA/Printing/Client Printers**

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto-create client printers</td>
<td>Auto-create all client printers</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Auto-create generic universal printer</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Client printer names</td>
<td>Standard printer names</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Direct connections to print servers</td>
<td>Enabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Printer driver mapping and compatibility</td>
<td>No rules are specified</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Printer properties retention</td>
<td>Held in profile only if not saved on client</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Retained and restored client printers</td>
<td>Allowed</td>
<td>XenDesktop 5, 5.5 and 5.6, Feature Pack 1</td>
</tr>
</tbody>
</table>
### ICA/Printing/Drivers

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic installation of in-box printer drivers</td>
<td>Enabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Universal driver preference</td>
<td>EMF; XPS; PCL5c; PCL4; PS</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Universal print driver usage</td>
<td>Use universal printing only if requested driver is unavailable</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### ICA/Printing/Universal Print Server

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Universal Print Server enable</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Universal Print Server print data stream (CGP) port</td>
<td>7229</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Universal Print Server print stream input bandwidth limit (kpbs)</td>
<td>0</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Universal Print Server web service (HTTP/SOAP) port</td>
<td>8080</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### ICA/Printing/Universal Printing

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Universal printing EMF processing mode</td>
<td>Spool directly to printer</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Universal printing image compression limit</td>
<td>Best quality (lossless compression)</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Universal printing optimization defaults</td>
<td>Image Compression</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td></td>
<td>• Desired image quality=Standard quality</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Enable heavyweight compression=False</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Image and Font Caching</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Allow caching of embedded images=True</td>
<td></td>
</tr>
</tbody>
</table>
### Universal Printing

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Universal printing preview preference</td>
<td>Do not use print preview for auto-created or generic universal printers</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Universal printing print quality limit</td>
<td>No limit</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### ICA/Security

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>SecureICA minimum encryption level</td>
<td>Basic</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
</tbody>
</table>

### ICA/Server Limits

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server idle timer interval</td>
<td>0 milliseconds</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
</tbody>
</table>

### ICA/Session Limits

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disconnected session timer</td>
<td>Disabled</td>
<td>XenDesktop 5, 5.5, 5.6 Feature Pack 1, 7 (Desktop OS), 7.1 (Desktop OS)</td>
</tr>
<tr>
<td>Disconnected session timer interval</td>
<td>1440 minutes</td>
<td>XenDesktop 5, 5.5, 5.6 Feature Pack 1, 7 (Desktop OS), 7.1 (Desktop OS)</td>
</tr>
<tr>
<td>Session connection timer</td>
<td>Disabled</td>
<td>XenDesktop 5, 5.5, 5.6 Feature Pack 1, 7 (Desktop OS), 7.1 (Desktop OS)</td>
</tr>
<tr>
<td>Session connection timer interval</td>
<td>1440 minutes</td>
<td>XenDesktop 5, 5.5, 5.6 Feature Pack 1, 7 (Desktop OS), 7.1 (Desktop OS)</td>
</tr>
<tr>
<td>Session idle timer</td>
<td>Enabled</td>
<td>XenDesktop 5, 5.5, 5.6 Feature Pack 1, 7 (Desktop OS), 7.1 (Desktop OS)</td>
</tr>
</tbody>
</table>
### ICA/Session Reliability

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session reliability connections</td>
<td>Allowed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Session reliability port number</td>
<td>2598</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Session reliability timeout</td>
<td>180 seconds</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### ICA/Time Zone Control

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimate local time for legacy clients</td>
<td>Enabled</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>Use local time of client</td>
<td>Use server time zone</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### ICA/TWAIN Devices

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client TWAIN device redirection</td>
<td>Allowed</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>TWAIN compression level</td>
<td>Medium</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
</tbody>
</table>

### ICA/USB Devices

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client USB device redirection</td>
<td>Prohibited</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>
### Client USB device redirection rules

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No rules are</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td></td>
<td>specified</td>
<td></td>
</tr>
<tr>
<td>Client USB Plug and Play device redirection</td>
<td>Allowed</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
</tbody>
</table>

### ICA/Visual Display

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target frame rate</td>
<td>30 fps</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Visual quality</td>
<td>Medium</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
</tbody>
</table>

### ICA/Visual Display/Moving Images

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum image quality</td>
<td>Normal</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Moving image compression</td>
<td>Enabled</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Progressive compression level</td>
<td>None</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Progressive compression threshold value</td>
<td>2147483647 Kbps</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Target minimum frame rate</td>
<td>10 fps</td>
<td>XenDesktop 5.5, 5.6 Feature Pack 1, 7 (Server OS and Desktop OS)</td>
</tr>
</tbody>
</table>

### ICA/Visual Display/Still Images

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extra color compression</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Name</td>
<td>Default setting</td>
<td>Applies to</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>Extra color compression threshold</td>
<td>8192 Kbps</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Heavyweight compression</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Lossy compression level</td>
<td>Medium</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Lossy compression threshold value</td>
<td>2147483647 Kbps</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

**ICA/WebSockets**

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebSockets connections</td>
<td>Prohibited</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>WebSockets port number</td>
<td>8008</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>WebSockets trusted origin server list</td>
<td>The wildcard, *, is used to trust all Receiver for Web URLs</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
</tbody>
</table>

**Load Management**

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concurrent logon tolerance</td>
<td>2</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>CPU usage</td>
<td>Disabled</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>CPU usage excluded process priority</td>
<td>Below Normal or Low</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>Disk usage</td>
<td>Disabled</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>Maximum number of sessions</td>
<td>250</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>Memory usage</td>
<td>Disabled</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
<tr>
<td>Memory usage base load</td>
<td>Zero load: 768MB</td>
<td>XenDesktop 7 (Server OS), 7.1 (Server OS)</td>
</tr>
</tbody>
</table>
### Profile Management/Advanced settings

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disable automatic configuration</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Log off user if a problem is encountered</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Number of retries when accessing locked files</td>
<td>5</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Process Internet cookie files on logoff</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### Profile Management/Basic settings

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active write back</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Enable Profile management</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Excluded groups</td>
<td>Disabled. Members of all user groups are processed.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Offline profile support</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Path to user store</td>
<td>Windows</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Process logons of local administrators</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Processed groups</td>
<td>Disabled. Members of all user groups are processed.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Name</td>
<td>Default setting</td>
<td>Applies to</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>--------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Cross-platform settings user groups</td>
<td>Disabled. All user groups specified in Processed groups the policy setting are processed</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Enable cross-platform settings</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Path to cross-platform definitions</td>
<td>Disabled. No path is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Path to cross-platform settings store</td>
<td>Disabled. Windows\PM_CMs is used.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Source for creating cross-platform settings</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

Profile Management/File System/Exclusions

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exclusion list - directories</td>
<td>Disabled. All folders in the user profile are synchronized.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Exclusion list - files</td>
<td>Disabled. All files in the user profile are synchronized.</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

Profile Management/File System/Synchronization

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories to synchronize</td>
<td>Disabled. Only non-excluded folders are synchronized.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Files to synchronize</td>
<td>Disabled. Only non-excluded files are synchronized.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Folders to mirror</td>
<td>Disabled. No folders are mirrored.</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

Profile Management/Folder Redirection

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
</table>

---

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved.
<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grant administrator access</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Include domain name</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### Profile Management/Folder Redirection/AppData(Roaming)

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppData(Roaming) path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Redirection settings for AppData(Roaming)</td>
<td>Contents are redirected to the UNC path specified in the AppData(Roaming) path policy settings</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### Profile Management/Folder Redirection/Contacts

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contacts path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Redirection settings for Contacts</td>
<td>Contents are redirected to the UNC path specified in the Contacts path policy settings</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### Profile Management/Folder Redirection/Desktop

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desktop path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Redirection settings for Desktop</td>
<td>Contents are redirected to the UNC path specified in the Desktop path policy settings</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### Profile Management/Folder Redirection/Documents

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documents path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Name</td>
<td>Default setting</td>
<td>Applies to</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Redirection settings for Documents</td>
<td>Contents are redirected to the UNC path specified in the Documents path policy settings</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Downloads path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Redirection settings for Downloads</td>
<td>Contents are redirected to the UNC path specified in the Downloads path policy settings</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Favorites path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Redirection settings for Favorites</td>
<td>Contents are redirected to the UNC path specified in the Favorites path policy settings</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Links path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Redirection settings for Links</td>
<td>Contents are redirected to the UNC path specified in the Links path policy settings</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Music path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>
### Profile Management/Folder Redirection/Pictures

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pictures path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Redirection settings for Pictures</td>
<td>Contents are redirected to the UNC path specified in the Pictures path policy settings</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### Profile Management/Folder Redirection/Saved Games

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Saved Games path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Redirection settings for Saved Games</td>
<td>Contents are redirected to the UNC path specified in the Saved Games path policy settings</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### Profile Management/Folder Redirection/Searches

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Searches path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Redirection settings for Searches</td>
<td>Contents are redirected to the UNC path specified in the Searches path policy settings</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### Profile Management/Folder Redirection/Start Menu

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Menu path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>
### Profile Management/Folder Redirection/Video

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video path</td>
<td>Disabled. No location is specified.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Redirection settings for Video</td>
<td>Contents are redirected to the UNC path specified in the Video path policy settings</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### Profile Management/Log settings

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active Directory actions</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Common information</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Common warnings</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Enable logging</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>File system actions</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>File system notifications</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Logoff</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Logon</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Name</td>
<td>Default setting</td>
<td>Applies to</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Maximum size of the log file</td>
<td>1048576</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Path to log file</td>
<td>Disabled. Log files are saved in the default location; %SystemRoot%\System32\LogFiles\UserProfileManager.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Personalized user information</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Policy values at logon and logoff</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Registry actions</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Registry differences at logoff</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

Profile Management/Profile handling

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay before deleting cached profiles</td>
<td>0</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Delete locally cached profiles on logoff</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Local profile conflict handling</td>
<td>Use local profile</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Migration of existing profiles</td>
<td>Local and roaming</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Path to the template profile</td>
<td>Disabled. New user profiles are created from the default user profile on the device where a user first logs on.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Template profile overrides local profile</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>
### Template profile overrides roaming profile

Disabled

### Profile Management/Registry

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exclusion list</td>
<td>Disabled. All registry keys in the HKCU hive are processed when a user logs off.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Inclusion list</td>
<td>Disabled. All registry keys in the HKCU hive are processed when a user logs off.</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### Profile Management/Streamed user profiles

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Always cache</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Always cache size</td>
<td>0 Mb</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Profile streaming</td>
<td>Disabled</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Streamed user profile groups</td>
<td>Disabled. All user profiles within an OU are processed normally.</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Timeout for pending area lock files (days)</td>
<td>1 day</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

### Receiver

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>StoreFront accounts</td>
<td>No stores are</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and</td>
</tr>
</tbody>
</table>
## Virtual Delivery Agent

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controller registration IPv6 netmask</td>
<td>No netmask is specified</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Controller registration port</td>
<td>80</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Controller SIDs</td>
<td>No SIDs are specified</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Controllers</td>
<td>No controllers are specified</td>
<td>All versions of XenDesktop</td>
</tr>
<tr>
<td>Enable auto update of controllers</td>
<td>Enabled</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Only use IPv6 controller registration</td>
<td>Disabled</td>
<td>XenDesktop 7 (Server OS and Desktop OS), 7.1 (Server OS and Desktop OS)</td>
</tr>
<tr>
<td>Site GUID</td>
<td>No GUID is specified</td>
<td>All versions of XenDesktop</td>
</tr>
</tbody>
</table>

## HDX 3D Pro

<table>
<thead>
<tr>
<th>Name</th>
<th>Default setting</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable lossless</td>
<td>Enabled</td>
<td>Applies to XenDesktop 5.5 and 5.6 Feature Pack 1</td>
</tr>
<tr>
<td>HDX 3D Pro quality settings</td>
<td></td>
<td>Applies to XenDesktop 5.5 and 5.6 Feature Pack 1</td>
</tr>
</tbody>
</table>
ICA policy settings

Dec 13, 2013
The ICA section contains policy settings related to ICA listener connections and mapping to the Clipboard.

Client clipboard redirection

This setting allows or prevents the Clipboard on the user device being mapped to the Clipboard on the server.

By default, clipboard redirection is allowed.

To prevent cut-and-paste data transfer between a session and the local Clipboard, select Prohibit. Users can still cut and paste data between applications running in sessions.

After allowing this setting, configure the maximum allowed bandwidth the Clipboard can consume in a client connection using the Clipboard redirection bandwidth limit or the Clipboard redirection bandwidth limit percent settings.

Desktop launches

This setting allows or prevents non-administrative users connecting to a desktop session on the server.

By default, non-administrative users cannot connect to these sessions.

ICA listener connection timeout

This setting specifies the maximum wait time for a connection using the ICA protocol to be completed.

By default, the maximum wait time is 120000 milliseconds, or two minutes.

ICA listener port number

This setting specifies the TCP/IP port number used by the ICA protocol on the server.

By default, the port number is set to 1494.

Valid port numbers must be in the range of 0–65535 and must not conflict with other well-known port numbers. If you change the port number, restart the server for the new value to take effect. If you change the port number on the server, you must also change it on every Receiver or plug-in that connects to the server.

Launching of non-published programs during client connection

This setting specifies whether to launch initial applications or published applications through ICA or RDP on the server.

By default, only published applications are allowed to launch.

Read-only clipboard

This setting enables or disables the copying and pasting of data from the session to the local user device.

By default, this setting is disabled and users can copy and paste data from the session to the local user device and from the local user device to the session.

When enabled, users can cut and paste data only from the local user device to the session.
When enabling this setting, make sure the Client clipboard redirection setting is present and set to Allowed. If this setting is disabled, the Clipboard on the user device is not mapped to the Clipboard on the server, and users cannot copy and paste data between the session and the local user device.
Audio policy settings

Dec 13, 2013
The Audio section contains policy settings you can configure to permit user devices to send and receive audio in sessions without reducing performance.

Audio over UDP real-time transport

This setting allows or prevents the transmission and receipt of audio between the VDA and user device over RTP using the User Datagram Protocol (UDP). When this setting is disabled, audio is sent and received over TCP.

By default, audio over UDP is allowed.

Audio Plug N Play

This setting allows or prevents the use of multiple audio devices to record and play sound.

By default, the use of multiple audio devices is allowed.

Audio quality

This setting specifies the quality level of sound received in user sessions.

By default, sound quality is set to High - high definition audio.

To control sound quality, choose one of the following options:

- Select Low - for low speed connections for low-bandwidth connections. Sounds sent to the client are compressed up to 16 Kbps. This compression results in a significant decrease in the quality of the sound but allows reasonable performance for a low-bandwidth connection.
- Select Medium - optimized for speech for most LAN-based connections. Sounds sent to the client are compressed up to 64 Kbps.
- Select High - high definition audio for connections where bandwidth is plentiful and sound quality is important. Clients can play sound at its native rate. Sounds can use up to 1.3 Mbps of bandwidth to play clearly. Transmitting this amount of data can result in increased CPU utilization and network congestion.

Bandwidth is consumed only while audio is recording or playing. If both occur at the same time, the bandwidth consumption is doubled.

To specify the maximum amount of bandwidth, configure the Audio redirection bandwidth limit or the Audio redirection bandwidth limit percent settings.

Client audio redirection

This setting specifies whether applications hosted on the server can play sounds through a sound device installed on the user device. This setting also specifies whether users can record audio input.

By default, audio redirection is allowed.

After allowing this setting, you can limit the bandwidth consumed by playing or recording audio. Limiting the amount of bandwidth consumed by audio can improve application performance but may also degrade audio quality. Bandwidth is consumed only while audio is recording or playing. If both occur at the same time, the bandwidth consumption doubles.
To specify the maximum amount of bandwidth, configure the Audio redirection bandwidth limit or the Audio redirection bandwidth limit percent settings.

Client microphone redirection

This setting enables or disables client microphone redirection. When enabled, users can use microphones to record audio input in a session.

By default, microphone redirection is allowed.

For security, users are alerted when servers that are not trusted by their devices try to access microphones. Users can choose to accept or not accept access. Users can disable the alert on Citrix Receiver.

If the Client audio redirection setting is disabled on the user device, this rule has no effect.
Auto Client Reconnect policy settings

Dec 13, 2013
The Auto Client Reconnect section contains policy settings for controlling automatic reconnection of sessions.

Auto client reconnect

This setting allows or prevents automatic reconnection by the same client after a connection has been interrupted.

By default, automatic reconnection is allowed.

Allowing automatic reconnection allows users to resume working where they were interrupted when a connection was broken. Automatic reconnection detects broken connections and then reconnects the users to their sessions.

However, automatic reconnection can result in a new session being launched (instead of reconnecting to an existing session) if Receiver's cookie, containing the key to the session ID and credentials, is not used. The cookie is not used if it has expired, for example, because of a delay in reconnection, or if credentials must be reentered. Auto client reconnect is not triggered if users intentionally disconnect.

Auto client reconnect authentication

This setting requires authentication for automatic client reconnections.

By default, authentication is not required.

When a user initially logs on, XenDesktop encrypts and stores the user credentials in memory and creates a cookie containing the encryption key which is sent to Receiver. When this setting is added, cookies are not used. Instead, XenDesktop displays a dialog box to users requesting credentials when the plug-in attempts to reconnect automatically.

Auto client reconnect logging

This setting enables or disables the recording of auto client reconnections in the event log.

By default, logging is disabled.

When logging is enabled, the server's System log captures information about successful and failed automatic reconnection events. A site does not provide a combined log of reconnection events for all servers.
Bandwidth policy settings

Dec 13, 2013

The Bandwidth section contains policy settings you can configure to avoid performance problems related to client session bandwidth use.

Important: Using these policy settings in conjunction with the Multi-Stream policy settings may produce unexpected results. If you use Multi-Stream settings in a policy, ensure these bandwidth limit policy settings are not included.

Audio redirection bandwidth limit

This setting specifies the maximum allowed bandwidth, in kilobits per second, for playing or recording audio in a user session.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the Audio redirection bandwidth limit percent setting, the most restrictive setting (with the lower value) is applied.

Audio redirection bandwidth limit percent

This setting specifies the maximum allowed bandwidth limit for playing or recording audio as a percentage of the total session bandwidth.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the Audio redirection bandwidth limit setting, the most restrictive setting (with the lower value) is applied.

If you configure this setting, you must also configure the Overall session bandwidth limit setting which specifies the total amount of bandwidth available for client sessions.

Client USB device redirection bandwidth limit

This setting specifies the maximum allowed bandwidth, in kilobits per second, for the redirection of USB devices to and from the client.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the Client USB device redirection bandwidth limit percent setting, the most restrictive setting (with the lower value) is applied.

Client USB device redirection bandwidth limit percent

This setting specifies the maximum allowed bandwidth for the redirection of USB devices to and from the client as a percentage of the total session bandwidth.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the Client USB device redirection bandwidth limit setting, the most restrictive setting (with the lower value) is applied.

If you configure this setting, you must also configure the Overall session bandwidth limit setting which specifies the total amount of bandwidth available for client sessions.
Clipboard redirection bandwidth limit

This setting specifies the maximum allowed bandwidth, in kilobits per second, for data transfer between a session and the local Clipboard.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the Clipboard redirection bandwidth limit percent setting, the most restrictive setting (with the lower value) is applied.

Clipboard redirection bandwidth limit percent

This setting specifies the maximum allowed bandwidth for data transfer between a session and the local Clipboard as a percentage of the total session bandwidth.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the Clipboard redirection bandwidth limit setting, the most restrictive setting (with the lower value) is applied.

If you configure this setting, you must also configure the Overall session bandwidth limit setting which specifies the total amount of bandwidth available for client sessions.

COM port redirection bandwidth limit

Note: For XenDesktop 7, configure this setting using the registry. For more information, see Configure COM Port and LPT Port Redirection settings using the registry.

This setting specifies the maximum allowed bandwidth in kilobits per second for accessing a COM port in a client connection. If you enter a value for this setting and a value for the COM port redirection bandwidth limit percent setting, the most restrictive setting (with the lower value) is applied.

COM port redirection bandwidth limit percent

Note: For XenDesktop 7, configure this setting using the registry. For more information, see Configure COM Port and LPT Port Redirection settings using the registry.

This setting specifies the maximum allowed bandwidth for accessing COM ports in a client connection as a percentage of the total session bandwidth.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the COM port redirection bandwidth limit setting, the most restrictive setting (with the lower value) is applied.

If you configure this setting, you must also configure the Overall session bandwidth limit setting which specifies the total amount of bandwidth available for client sessions.

File redirection bandwidth limit

This setting specifies the maximum allowed bandwidth, in kilobits per second, for accessing a client drive in a user session.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the File redirection bandwidth limit percent setting, the most restrictive setting (with the lower value) takes effect.
File redirection bandwidth limit percent

This setting specifies the maximum allowed bandwidth limit for accessing client drives as a percentage of the total session bandwidth.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the File redirection bandwidth limit setting, the most restrictive setting (with the lower value) is applied.

If you configure this setting, you must also configure the Overall session bandwidth limit setting which specifies the total amount of bandwidth available for client sessions.

HDX MediaStream Multimedia Acceleration bandwidth limit

This setting specifies the maximum allowed bandwidth limit, in kilobits per second, for delivering streaming audio and video using HDX MediaStream Multimedia Acceleration.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the HDX MediaStream Multimedia Acceleration bandwidth limit percent setting, the most restrictive setting (with the lower value) takes effect.

HDX MediaStream Multimedia Acceleration bandwidth limit percent

This setting specifies the maximum allowed bandwidth for delivering streaming audio and video using HDX MediaStream Multimedia Acceleration as a percentage of the total session bandwidth.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the HDX MediaStream Multimedia Acceleration bandwidth limit setting, the most restrictive setting (with the lower value) takes effect.

If you configure this setting, you must also configure the Overall session bandwidth limit setting which specifies the total amount of bandwidth available for client sessions.

LPT port redirection bandwidth limit

Note: For XenDesktop 7, configure this setting using the registry. For more information, see Configure COM Port and LPT Port Redirection settings using the registry.

This setting specifies the maximum allowed bandwidth, in kilobits per second, for print jobs using an LPT port in a single user session.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the LPT port redirection bandwidth limit percent setting, the most restrictive setting (with the lower value) is applied.

LPT port redirection bandwidth limit percent

Note: For XenDesktop 7, configure this setting using the registry. For more information, see Configure COM Port and LPT Port Redirection settings using the registry.

This setting specifies the bandwidth limit for print jobs using an LPT port in a single client session as a percentage of the total session bandwidth.
By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the LPT port redirection bandwidth limit setting, the most restrictive setting (with the lower value) is applied.

If you configure this setting, you must also configure the Overall session bandwidth limit setting which specifies the total amount of bandwidth available for client sessions.

Overall session bandwidth limit
This setting specifies the total amount of bandwidth available, in kilobits per second, for user sessions.

By default, no maximum (zero) is specified.

Limiting the amount of bandwidth consumed by a client connection can improve performance when other applications outside the client connection are competing for limited bandwidth.

Printer redirection bandwidth limit
This setting specifies the maximum allowed bandwidth, in kilobits per second, for accessing client printers in a user session.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the Printer redirection bandwidth limit percent setting, the most restrictive setting (with the lower value) is applied.

Printer redirection bandwidth limit percent
This setting specifies the maximum allowed bandwidth for accessing client printers as a percentage of the total session bandwidth.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the Printer redirection bandwidth limit percent setting, the most restrictive setting (with the lower value) is applied.

If you configure this setting, you must also configure the Overall session bandwidth limit setting which specifies the total amount of bandwidth available for client sessions.

TWAIN device redirection bandwidth limit
This setting specifies the maximum allowed bandwidth, in kilobits per second, for controlling TWAIN imaging devices from published applications.

By default, no maximum (zero) is specified.

If you enter a value for this setting and a value for the TWAIN device redirection bandwidth limit percent setting, the most restrictive setting (with the lower value) is applied.

TWAIN device redirection bandwidth limit percent
This setting specifies the maximum allowed bandwidth for controlling TWAIN imaging devices from published applications as a percentage of the total session bandwidth.

By default, no maximum (zero) is specified.
If you enter a value for this setting and a value for the TWAIN device redirection bandwidth limit setting, the most restrictive setting (with the lower value) is applied.

If you configure this setting, you must also configure the Overall session bandwidth limit setting which specifies the total amount of bandwidth available for client sessions.
Client Sensors policy settings

Dec 13, 2013
The Client Sensors section contains policy settings for controlling how mobile device sensor information is handled in a user session.

Allow applications to use the physical location of the client device

This setting determines whether applications running in a session on a mobile device are allowed to use the physical location of the user device.

By default, the use of location information is prohibited

When this setting is prohibited, attempts by an application to retrieve location information return a “permission denied” value.

When this setting is allowed, a user can prohibit use of location information by denying a Receiver request to access the location. Android and iOS devices prompt at the first request for location information in each session.

When developing hosted applications that use the Allow applications to use the physical location of the client device setting, consider the following:

● A location-enabled application should not rely on location information being available because:
  ● A user might not allow access to location information.
  ● The location might not be available or might change while the application is running.
  ● A user might connect to the application session from a different device that does not support location information.
● A location-enabled application must:
  ● Have the location feature off by default.
  ● Provide a user option to allow or disallow the feature while the application is running.
  ● Provide a user option to clear location data that is cached by the application. (Receiver does not cache location data.)
● A location-enabled application must manage the granularity of the location information so that the data acquired is appropriate to the purpose of the application and conforms to regulations in all relevant jurisdictions.
● A secure connection (for example, using SSL/TLS or a VPN) should be enforced when using location services. Citrix Receiver should connect to trusted servers.
● Consider obtaining legal advice regarding the use of location services.
Desktop UI policy settings

Dec 13, 2013

The Desktop UI section contains policy settings that control visual effects, such as desktop wallpaper, menu animations, and drag-and-drop images, to manage the bandwidth used in client connections. You can improve application performance on a WAN by limiting bandwidth usage.

Desktop Composition Redirection

This setting specifies whether to use the processing capabilities of the graphics processing unit (GPU) or integrated graphics processor (IGP) on the user device for local DirectX graphics rendering to provide users with a more fluid Windows desktop experience. When enabled, Desktop Composition Redirection delivers a highly responsive Windows experience while maintaining high scalability on the server.

By default, Desktop Composition Redirection is enabled.

To turn off Desktop Composition Redirection and reduce the bandwidth required in user sessions, select Disabled when adding this setting to a policy.

Desktop Composition Redirection graphics quality

This setting specifies the quality of graphics used for Desktop Composition Redirection.

By default, this is set to high.

Choose from High, Medium, Low, or Lossless quality.

Desktop wallpaper

This setting allows or prevents wallpaper showing in user sessions.

By default, user sessions can show wallpaper.

To turn off desktop wallpaper and reduce the bandwidth required in user sessions, select Prohibited when adding this setting to a policy.

Menu animation

This setting allows or prevents menu animation in user sessions.

By default, menu animation is allowed.

Menu animation is a Microsoft personal preference setting that causes a menu to appear after a short delay, either by scrolling or fading in. When this policy setting is set to Allowed, an arrow icon appears at the bottom of the menu. The menu appears when you point to that arrow.

View window contents while dragging

This setting allows or prevents the display of window contents when dragging a window across the screen.

By default, viewing window contents is allowed.

When set to Allowed, the entire window appears to move when you drag it. When set to Prohibited, only the window...
outline appears to move until you drop it.
End User Monitoring policy settings

Dec 13, 2013
The End User Monitoring section contains policy settings for measuring session traffic.

ICA round trip calculation
This setting determines whether ICA round trip calculations are performed for active connections.
By default, calculations for active connections are enabled.
By default, each ICA round trip measurement initiation is delayed until some traffic occurs that indicates user interaction. This delay can be indefinite in length and is designed to prevent the ICA round trip measurement being the sole reason for ICA traffic.

ICA round trip calculation interval
This setting specifies the frequency, in seconds, at which ICA round trip calculations are performed.
By default, ICA round trip is calculated every 15 seconds.

ICA round trip calculations for idle connections
This setting determines whether ICA round trip calculations are performed for idle connections.
By default, calculations are not performed for idle connections.
By default, each ICA round trip measurement initiation is delayed until some traffic occurs that indicates user interaction. This delay can be indefinite in length and is designed to prevent the ICA round trip measurement being the sole reason for ICA traffic.
Enhanced Desktop Experience policy settings

Dec 13, 2013

The Enhanced Desktop Experience section contains policy settings for configuring virtual desktops to look like local Windows 7 desktops.

Enhanced Desktop Experience

This setting configures sessions running on server operating systems to look like local Windows 7 desktops, providing users with an enhanced desktop experience.

By default, this setting is allowed and sessions running on server operating systems are configured to look like local Windows 7 desktops.

If a user profile with Windows Classic theme already exists on the virtual desktop, enabling this policy does not provide an enhanced desktop experience for that user. If a user, with a Windows 7 theme user profile, logs on to a virtual desktop running Windows Server 2012 for which this policy is either not configured or disabled, an error message, indicating failure to apply the theme is shown to that user.

In both cases resetting the user profile resolves the issue.

If the policy switches from enabled to disabled on a virtual desktop with active user sessions, the look and feel of those sessions is inconsistent with both the Windows 7 and Windows Classic desktop experience. To avoid this, ensure you restart the virtual desktop after changing the state of this policy setting. You must also delete any roaming profiles on the virtual desktop. Citrix also recommends deleting any other user profiles on the virtual desktop to avoid inconsistencies between profiles.

If you are using roaming user profiles in your environment, ensure the Enhanced Desktop Experience feature is enabled or disabled for all virtual desktops that share a profile.

Citrix does not recommend sharing roaming profiles between virtual desktops running server operating systems and client operating systems. Profiles for client and server operating systems differ and sharing roaming profiles across both types can lead to inconsistencies in profile properties when a user moves between the two.
File Redirection policy settings

Dec 13, 2013

The File Redirection section contains policy settings relating to client drive mapping and client drive optimization.

Auto connect client drives

This setting allows or prevents automatic connection of client drives when users log on.

By default, automatic connection is allowed.

When adding this setting to a policy, make sure you enable the settings for the drive types you want automatically connected. For example, to allow automatic connection of users' CD-ROM drives, configure this setting and the Client optical drives setting.

Related policy settings

- Client drive redirection
- Client floppy drives
- Client optical drives
- Client fixed drives
- Client network drives
- Client removable drives

Client drive redirection

This setting enables or disables file redirection to and from drives on the user device.

By default, file redirection is enabled.

When enabled, users can save files to all their client drives. When disabled, all file redirection is prevented, regardless of the state of the individual file redirection settings such as Client floppy drives and Client network drives.

Related policy settings

- Client floppy drives
- Client optical drives
- Client fixed drives
- Client network drives
- Client removable drives

Client fixed drives

This setting allows or prevents users from accessing or saving files to fixed drives on the user device.

By default, accessing client fixed drives is allowed.

When adding this setting to a policy, make sure the Client drive redirection setting is present and set to Allowed. If these settings are disabled, client fixed drives are not mapped and users cannot access these drives manually, regardless of the state of the Client fixed drives setting.

To ensure fixed drives are automatically connected when users log on, configure the Auto connect client drives setting.
Client floppy drives
This setting allows or prevents users from accessing or saving files to floppy drives on the user device.

By default, accessing client floppy drives is allowed.

When adding this setting to a policy, make sure the Client drive redirection setting is present and set to Allowed. If these settings are disabled, client floppy drives are not mapped and users cannot access these drives manually, regardless of the state of the Client floppy drives setting.

To ensure floppy drives are automatically connected when users log on, configure the Auto connect client drives setting.

Client network drives
This setting allows or prevents users from accessing and saving files to network (remote) drives through the user device.

By default, accessing client network drives is allowed.

When adding this setting to a policy, make sure the Client drive redirection setting is present and set to Allowed. If these settings are disabled, client network drives are not mapped and users cannot access these drives manually, regardless of the state of the Client network drives setting.

To ensure network drives are automatically connected when users log on, configure the Auto connect client drives setting.

Client optical drives
This setting allows or prevents users from accessing or saving files to CD-ROM, DVD-ROM, and BD-ROM drives on the user device.

By default, accessing client optical drives is allowed.

When adding this setting to a policy, make sure the Client drive redirection setting is present and set to Allowed. If these settings are disabled, client optical drives are not mapped and users cannot access these drives manually, regardless of the state of the Client optical drives setting.

To ensure optical drives are automatically connected when users log on, configure the Auto connect client drives setting.

Client removable drives
This setting allows or prevents users from accessing or saving files to USB drives on the user device.

By default, accessing client removable drives is allowed.

When adding this setting to a policy, make sure the Client drive redirection setting is present and set to Allowed. If these settings are disabled, client removable drives are not mapped and users cannot access these drives manually, regardless of the state of the Client removable drives setting.

To ensure removable drives are automatically connected when users log on, configure the Auto connect client drives setting.

Host to client redirection
This setting enables or disables file type associations for URLs and some media content to be opened on the user device. When disabled, content opens on the server.
By default, file type association is disabled.

These URL types are opened locally when you enable this setting:
- Hypertext Transfer Protocol (HTTP)
- Secure Hypertext Transfer Protocol (HTTPS)
- Real Player and QuickTime (RTSP)
- Real Player and QuickTime (RTSPU)
- Legacy Real Player (PNM)
- Microsoft Media Server (MMS)

Preserve client drive letters

This setting enables or disables mapping of client drives to the same drive letter in the session.

By default, client drive letters are not preserved.

When adding this setting to a policy, make sure the Client drive redirection setting is present and set to Allowed.

Read-only client drive access

This setting allows or prevents users and applications from creating or modifying files or folders on mapped client drives.

By default, files and folders on mapped client drives can be modified.

If set to Enabled, files and folders are accessible with read-only permissions.

When adding this setting to a policy, make sure the Client drive redirection setting is present and set to Allowed.

Special folder redirection

This setting allows or prevents Citrix Receiver and Web Interface users to see their local Documents and Desktop special folders from a session.

By default, special folder redirection is allowed.

This setting prevents any objects filtered through a policy from having special folder redirection, regardless of settings that exist elsewhere. When you allow this setting, any related settings specified for the Web Interface or Citrix Receiver are ignored.

To define which users can have special folder redirection, select Allowed and include this setting in a policy filtered on the users you want to have this feature. This setting overrides all other special folder redirection settings.

Because special folder redirection must interact with the user device, policy settings that prevent users from accessing or saving files to their local hard drives also prevent special folder redirection from working.

When adding this setting to a policy, make sure the Client fixed drives setting is present and set to Allowed.

Use asynchronous writes

This setting enables or disables asynchronous disk writes.

By default, asynchronous writes are disabled.

Asynchronous disk writes can improve the speed of file transfers and writing to client disks over WANs, which are typically
characterized by relatively high bandwidth and high latency. However, if there is a connection or disk fault, the client file or files being written may end in an undefined state. If this happens, a pop-up window informs the user of the files affected. The user can then take remedial action, such as restarting an interrupted file transfer on reconnection or when the disk fault is corrected.

Citrix recommends enabling asynchronous disk writes only for users who need remote connectivity with good file access speed and who can easily recover files or data lost in the event of connection or disk failure.

When adding this setting to a policy, make sure that the Client drive redirection setting is present and set to Allowed. If this setting is disabled, asynchronous writes will not occur.
The Flash Redirection section contains policy settings for handling Flash content in user sessions.

Flash acceleration

This setting enables or disables Flash content rendering on user devices instead of the server. By default, client-side Flash content rendering is enabled.

Note: This setting is used for legacy Flash redirection with Citrix online plug-in 12.1.

When enabled, this setting reduces network and server load by rendering Flash content on the user device. Additionally, the Flash URL compatibility list setting forces Flash content from specific websites to be rendered on the server.

On the user device, the Enable HDX MediaStream for Flash on the user device setting must be enabled as well.

When this setting is disabled, Flash content from all websites, regardless of URL, is rendered on the server. To allow only certain websites to render Flash content on the user device, configure the Flash URL compatibility list setting.

Flash background color list

This setting enables you to set key colors for given URLs.

By default, no key colors are specified.

Key colors appear behind client-rendered Flash and help provide visible region detection. The key color specified should be rare; otherwise, visible region detection might not work properly.

Valid entries consist of a URL (with optional wildcards at the beginning or end) followed by a 24-bit RGB color hexadecimal code. For example: http://citrix.com 000003

Flash backwards compatibility

This setting enables or disables the use of original, legacy Flash redirection features with older versions of Citrix Receiver (formerly the Citrix online plug-in).

By default, this setting is enabled.

On the user device, the Enable HDX MediaStream for Flash on the user device setting must be enabled as well.

Second generation Flash redirection features are enabled for use with Citrix Receiver 3.0. Legacy redirection features are supported for use with the Citrix online plug-in 12.1. To ensure second generation Flash redirection features are used, both the server and the user device must have second generation Flash redirection enabled. If legacy redirection is enabled on either the server or the user device, legacy redirection features are used.

Flash default behavior

This setting establishes the default behavior for second generation Flash acceleration.

By default, Flash acceleration is enabled.

To configure this setting, choose one of the following options:
Enable Flash acceleration—Flash Redirection is used.
Block Flash Player—Flash Redirection and server-side rendering are not used. The user cannot view any Flash content.
Disable Flash acceleration—Flash Redirection is not used. The user can view server-side rendered Flash content if a version of Adobe Flash Player for Windows Internet Explorer compatible with the content is installed on the server.

This setting can be overridden for individual Web pages and Flash instances based on the configuration of the Flash URL compatibility list setting. Additionally, the user device must have the Enable HDX MediaStream for Flash on the user device setting enabled.

Flash event logging
This setting enables Flash events to be recorded in the Windows application event log.

By default, logging is allowed.

On computers running Windows 7 or Windows Vista, a Flash redirection-specific log appears in the Applications and Services Log node.

Flash intelligent fallback
This setting enables or disables automatic attempts to employ server-side rendering for Flash Player instances where client-side rendering is either unnecessary or provides a poor user experience.

By default, this setting is enabled.

Flash latency threshold
This setting specifies a threshold between 0-30 milliseconds to determine where Adobe Flash content is rendered.

By default, the threshold is 30 milliseconds.
During startup, HDX MediaStream for Flash measures the current latency between the server and user device. If the latency is under the threshold, HDX MediaStream for Flash is used to render Flash content on the user device. If the latency is above the threshold, the network server renders the content if an Adobe Flash player is available there.

When enabling this setting, make sure the Flash backwards compatibility setting is also present and set to Enabled.

Note: Applies only when using HDX MediaStream Flash redirection in Legacy mode.

Flash server-side content fetching URL list
This setting specifies websites whose Flash content can be downloaded to the server and then transferred to the user device for rendering.

By default, no sites are specified.
This setting is used when the user device does not have direct access to the Internet; the server provides that connection. Additionally, the user device must have the Enable server-side content fetching setting enabled.

Second generation Flash redirection includes a fallback to server-side content fetching for Flash .swf files. If the user device is unable to fetch Flash content from a Web site, and the Web site is specified in the Flash server-side content fetching URL list, server-side content fetching occurs automatically.
When adding URLs to the list:
- Add the URL of the Flash application instead of the top-level HTML page that initiates the Flash Player.
● Use an asterisk (*) at the beginning or end of the URL as a wildcard.
● Use a trailing wildcard to allow all child URLs (http://www.citrix.com/*).
● The prefixes http:// and https:// are used when present, but are not required for valid list entries.

Flash URL compatibility list

This setting specifies the rules which determine whether Flash content on certain websites are rendered on the user device, rendered on the server, or blocked from rendering.

By default, no rules are specified.

When adding URLs to the list:
● Prioritize the list with the most important URLs, actions, and rendering locations at the top.
● Use an asterisk (*) at the beginning or end of the URL as a wildcard.
● Use a trailing wildcard to refer to all child URLs (http://www.citrix.com/*).
● The prefixes http:// and https:// are used when present, but are not required for valid list entries.
● Add to this list websites whose Flash content does not render correctly on the user device and select either the Render on Server or Block options.
The Graphics section contains policy settings for controlling how images are handled in user sessions.

Display memory limit
This setting specifies the maximum video buffer size in kilobytes for the session.
By default, the display memory limit is 32768 kilobytes.
Specify an amount in kilobytes from 128 to 131072. Using more color depth and higher resolution for connections requires more memory. If the memory limit is reached, the display degrades according to the Display mode degrade preference setting.

Display mode degrade preference
Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled.
This setting specifies whether color depth or resolution degrades first when the session display memory limit is reached.
By default, color depth is degraded first.
When the session memory limit is reached, you can reduce the quality of displayed images by choosing whether color depth or resolution is degraded first. When color depth is degraded first, displayed images use fewer colors. When resolution is degraded first, displayed images use fewer pixels per inch.
To notify users when either color depth or resolution are degraded, configure the Notify user when display mode is degraded setting.

Dynamic windows preview
This setting enables or disables the display of seamless windows in Flip, Flip 3D, Taskbar Preview, and Peek window preview modes.
By default, this setting is enabled.

Image caching
Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled.
This setting enables or disables caching of images in sessions. When needed, the images are retrieved in sections to make scrolling smoother.
By default, image caching is enabled.

Legacy graphics mode
Applies to XenDesktop 7.0 (Server OS and Desktop OS).
This setting disables the rich graphics experience, providing fallback to the legacy graphics experience to improve scalability over a WAN or mobile connection.
By default, this setting is disabled and users are provided with the rich graphics experience.
Maximum allowed color depth

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled. This setting specifies the maximum color depth allowed for a session.

By default, the maximum allowed color depth is 32 bits per pixel.

This setting applies only to ThinWire drivers and connections. It does not apply to VDAs that have a non-ThinWire driver as the primary display driver, such as VDAs that use a Windows Display Driver Model (WDDM) driver as the primary display driver. For Desktop OS VDAs using a WDDM driver as the primary display driver, this setting has no effect. For Windows Server OS VDAs using a WDDM driver, this setting might prevent users from connecting to the VDA.

Setting a high color depth requires more memory. To degrade color depth when the memory limit is reached, configure the Display mode degrade preference setting. When color depth is degraded, displayed images use fewer colors.

Notify user when display mode is degraded

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled. This setting displays a brief explanation to the user when the color depth or resolution is degraded.

By default, notifying users is disabled.

Queuing and tossing

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled. This setting discards queued images that are replaced by another image.

By default, queuing and tossing is enabled.

This improves response when graphics are sent to the user device. Configuring this setting can cause animations to become choppy due to dropped frames.
Caching policy settings

Dec 13, 2013
The Caching section contains policy settings that enable you to cache image data on user devices when client connections are limited in bandwidth.

Persistent Cache threshold

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled. This setting caches bitmaps on the hard drive of the user device. This enables re-use of large, frequently-used images from previous sessions.

By default, the threshold is 3000000 bits per second.

The threshold value represents the point below which you want the Persistent Cache feature to take effect. For example, with regard to the default value, bitmaps are cached on the hard drive of the user device when bandwidth is below 3000000 bps.
Keep Alive policy settings

Dec 13, 2013
The Keep Alive section contains policy settings for managing ICA keep-alive messages.

ICA keep alive timeout

This setting specifies the number of seconds between successive ICA keep-alive messages.

By default, the interval between keep-alive messages is 60 seconds.

Specify an interval between 1-3600 seconds in which to send ICA keep-alive messages. Do not configure this setting if your network monitoring software is responsible for closing inactive connections.

ICA keep alives

This setting enables or disables sending ICA keep-alive messages periodically.

By default, keep-alive messages are not sent.

Enabling this setting prevents broken connections from being disconnected. If the server detects no activity, this setting prevents Remote Desktop Services (RDS) from disconnecting the session. The server sends keep-alive messages every few seconds to detect if the session is active. If the session is no longer active, the server marks the session as disconnected.

ICA Keep-Alive does not work if you are using Session Reliability. Configure ICA Keep-Alive only for connections that are not using Session Reliability.

Related policy settings

Session reliability connections
Local App Access policy settings

Dec 13, 2013

The Local App Access section contains policy settings you can configure to enable the integration of users' locally-installed applications with hosted applications within a hosted desktop environment.

Allow local app access

This setting allows or prevents the integration of users' locally-installed applications with hosted applications within a hosted desktop environment.

When a user launches a locally-installed application, that application appears to run within their virtual desktop, even though it is actually running locally.

By default, local app access is prohibited.

URL redirection black list

This setting specifies websites that are redirected to and launched in the local Web browser. This might include websites requiring locale information, such as msn.com or newsgoogle.com, or websites containing rich media content that are better rendered on the user device.

By default, no sites are specified.

URL redirection white list

This setting specifies websites that are rendered in the environment in which they are launched.

By default, no sites are specified.
Mobile Experience policy settings

Dec 13, 2013
The Mobile Experience section contains policy settings for handling the Citrix Mobility Pack.

Automatic keyboard display

This setting enables or disables the automatic display of the keyboard on mobile device screens.

By default, the automatic display of the keyboard is disabled.

Launch touch-optimized desktop

This setting determines the overall Receiver interface behavior by allowing or prohibiting a touch-friendly interface that is optimized for tablet devices.

By default, a touch-friendly interface is used.

To use only the Windows interface, set this policy to Prohibited.

Remote the combo box

This setting determines the types of combo boxes you can display in sessions on mobile devices. To display the device-native combo box control, set this policy to Allowed. When this setting is allowed, a user can change a Receiver for iOS session setting to use the Windows combo box.

By default, the Remote the combo box feature is prohibited.
Multimedia policy settings

Dec 13, 2013

The Multimedia section contains policy settings for managing streaming audio and video in user sessions.

Limit video quality

This setting specifies the maximum video quality level allowed for an HDX connection. When configured, maximum video quality is limited to the specified value, ensuring that multimedia Quality of Service (QoS) is maintained within an environment.

By default, this setting is not configured.

To limit the maximum video quality level allowed, choose one of the following options:

- 1080p/8.5mbps
- 720p/4.0mbps
- 480p/720kbps
- 380p/400kbps
- 240p/200kbps

Note: Playing multiple videos simultaneously on the same server consumes large amounts of resources and may impact server scalability.

Multimedia conferencing

This setting allows or prevents support for video conferencing applications.

By default, video conferencing support is allowed.

When adding this setting to a policy, make sure the Windows Media Redirection setting is present and set to Allowed.

When using multimedia conferencing, make sure the following conditions are met:

- Manufacturer-supplied drivers for the web cam used for multimedia conferencing must be installed.
- The web cam must be connected to the user device before initiating a video conferencing session. The server uses only one installed web cam at any given time. If multiple webcams are installed on the user device, the server attempts to use each web cam in succession until a video conferencing session is created successfully.
- An Office Communicator server must be present in your environment.
- The Office Communicator client software must be published on the server.

Optimization for Windows Media multimedia redirection over WAN

This setting enables real-time multimedia transcoding, allowing audio and video media streaming to mobile devices, and enhancing the user experience by improving how Windows Media content is delivered over a WAN.

By default, the delivery of Windows Media content over the WAN is optimized.

When adding this setting to a policy, make sure the Windows Media Redirection setting is present and set to Allowed.

When this setting is enabled, real-time multimedia transcoding is deployed automatically as needed to enable media streaming, providing a seamless user experience even in extreme network conditions.

Use GPU for optimizing Windows Media multimedia redirection over WAN
This setting enables real-time multimedia transcoding to be done in the Graphics Processing Unit (GPU) on the Delivery Agent, to improve server scalability. GPU transcoding is available only if the Delivery Agent has a supported GPU for hardware acceleration. Otherwise, transcoding falls back to the CPU.

Note: GPU transcoding is supported only on NVIDIA GPUs.

By default, using the GPU on the Delivery Agent to optimize the delivery of Windows Media content over the WAN is prohibited.

When adding this setting to a policy, make sure the Windows Media Redirection and Optimization for Windows Media multimedia redirection over WAN settings are present and set to Allowed.

Windows Media client-side content fetching

This setting enables a user device to stream multimedia files directly from the source provider on the Internet or Intranet, rather than through the host server.

By default, the streaming of multimedia files to the user device direct from the source provider is allowed.

Allowing this setting improves network utilization and server scalability by moving any processing on the media from the host server to the user device. It also removes the requirement that an advanced multimedia framework such as Microsoft DirectShow or Media Foundation be installed on the user device; the user device requires only the ability to play a file from a URL.

When adding this setting to a policy, make sure the Windows Media Redirection setting is present and set to Allowed. If this setting is disabled, the streaming of multimedia files to the user device direct from the source provider is also disabled.

Windows Media Redirection

This setting controls and optimizes the way servers deliver streaming audio and video to users.

By default, the delivery of streaming audio and video to users is allowed.

Allowing this setting increases the quality of audio and video rendered from the server to a level that compares with audio and video played locally on a user device. The server streams multimedia to the client in the original, compressed form and allows the user device to decompress and render the media.

Windows Media redirection optimizes multimedia files that are encoded with codecs that adhere to Microsoft’s DirectShow, DirectX Media Objects (DMO), and Media Foundation standards. To play back a given multimedia file, a codec compatible with the encoding format of the multimedia file must be present on the user device.

By default, audio is disabled on Citrix Receiver. To allow users to run multimedia applications in ICA sessions, turn on audio or give the users permission to turn on audio themselves in their Receiver interface.

Select Prohibited only if playing media using Windows Media redirection appears worse than when rendered using basic ICA compression and regular audio. This is rare but can happen under low bandwidth conditions; for example, with media in which there is a very low frequency of key frames.

Windows Media Redirection buffer size

This setting specifies a buffer size from 1 to 10 seconds for multimedia acceleration.

By default, the buffer size is 5 seconds.
Windows Media Redirection buffer size use

This setting enables or disables using the buffer size specified in the Windows Media Redirection buffer size setting.

By default, the buffer size specified is not used.

If this setting is disabled or if the Windows Media Redirection buffer size setting is not configured, the server uses the default buffer size value (5 seconds).
Multi-Stream Connections policy settings

Dec 13, 2013
The Multi-Stream Connections section contains policy settings for managing Quality of Service (QoS) prioritization for multiple ICA connections in a session.

Audio over UDP

This setting allows or prevents audio over UDP on the server.

By default, audio over UDP is allowed on the server.

When enabled, this setting opens a UDP port on the server to support all connections configured to use Audio over UDP Realtime Transport.

Audio UDP port range

This setting specifies the range of port numbers used by the VDA to exchange audio packet data with the user device.

By default, this is set to 16500,16509.

The VDA attempts to use each UDP port pair to exchange data with the user device, starting with the lowest and incrementing by 2 for each subsequent attempt. Each specified port handles both inbound and outbound traffic.

Enter a range, in the format lowest port number, highest port number.

Multi-Port policy

This setting specifies the TCP ports to be used for ICA traffic and establishes the network priority for each port.

By default, the primary port (2598) has a High priority.

When you configure ports, you can assign the following priorities:

- Very High
- High
- Medium
- Low

You might assign a Very High priority when real-time responsiveness is required, such as for audio and video conferencing. As well, you might assign a Low priority to background processes such as printing. Each port must have a unique priority. For example, you cannot assign a Very High priority to both CGP port 1 and CGP port 3.

To remove a port from prioritization, set the port number to 0. You cannot remove the primary port and you cannot modify its priority level.

When configuring this setting, reboot the server. This setting takes effect only when the Multi-Stream computer setting policy setting is enabled.

Multi-Stream computer setting

This setting enables or disables Multi-Stream on the server.
By default, Multi-Stream is disabled.

If you use Citrix Cloudbridge with Multi-Stream support in your environment, you do not need to configure this setting. Configure this policy setting when using third-party routers or legacy Branch Repeaters to achieve the desired Quality of Service.

When configuring this setting, reboot the server to ensure changes take effect.
Important: Using this policy setting in conjunction with bandwidth limit policy settings such as Overall session bandwidth limit may produce unexpected results. When including this setting in a policy, ensure that bandwidth limit settings are not included.

Multi-Stream user setting

This setting enables or disables Multi-Stream on the user device.

By default, Multi-Stream is disabled for all users.

This setting takes effect only on hosts where the Multi-Stream computer setting policy setting is enabled.
Important: Using this policy setting in conjunction with bandwidth limit policy settings such as Overall session bandwidth limit may produce unexpected results. When including this setting in a policy, ensure that bandwidth limit settings are not included.
Port Redirection policy settings

Dec 13, 2013
The Port Redirection section contains policy settings for client LPT and COM port mapping.

Note: For XenDesktop 7, configure these setting using the registry. For more information, see Configure COM Port and LPT Port Redirection settings using the registry.

Auto connect client COM ports

This setting enables or disables automatic connection of COM ports on user devices when users log on to a site.

By default, client COM ports are not automatically connected.

Auto connect client LPT ports

This setting enables or disables automatic connection of LPT ports on user devices when users log on to a site.

By default, client LPT ports are connected automatically.

Client COM port redirection

This setting allows or prevents access to COM ports on the user device.

By default, COM port redirection is prohibited.

Related policy settings
- COM port redirection bandwidth limit
- COM port redirection bandwidth limit percent

Client LPT port redirection

This setting allows or prevents access to LPT ports on the user device.

By default, LPT port redirection is prohibited.

LPT ports are used only by legacy applications that send print jobs to the LPT ports and not to the print objects on the user device. Most applications today can send print jobs to printer objects. This policy setting is necessary only for servers that host legacy applications that print to LPT ports.

Related policy settings
- LPT port redirection bandwidth limit
- LPT port redirection bandwidth limit percent
Printing policy settings

Dec 13, 2013
The Printing section contains policy settings for managing client printing.

Client printer redirection

This setting controls whether or not client printers are mapped to a server when a user logs on to a session.

By default, client printer mapping is allowed.

Related Policy Settings
Auto-create client printers

Default printer

This setting specifies how the default printer on the user device is established in a session.

By default, the user's current printer is used as the default printer for the session.

To use the current Remote Desktop Services or Windows user profile setting for the default printer, select Do not adjust the user's default printer. If you choose this option, the default printer is not saved in the profile and it does not change according to other session or client properties. The default printer in a session will be the first printer auto-created in the session, which is either:
- The first printer added locally to the Windows server in Control Panel > Devices and Printers.
- The first auto-created printer, if there are no printers added locally to the server.

You can use this option to present users with the nearest printer through profile settings (known as Proximity Printing).

Printer assignments

This setting provides an alternative to the Default printer and Session printers settings. Use the individual Default printer and Session printers settings to configure behaviors for a site, large group, or organizational unit. Use the Printer assignments setting to assign a large group of printers to multiple users.

This setting specifies how the default printer on the listed user devices is established in a session.

By default, the user's current printer is used as the default printer for the session.

It also specifies the network printers to be auto-created in a session for each user device. By default, no printers are specified.

- When setting the default printer value:
  - To use the current default printer for the user device, select Do not adjust.
  - To use the current Remote Desktop Services or Windows user profile setting for the default printer, select Do no adjust. If you choose this option, the default printer is not saved in the profile and it does not change according to other session or client properties. The default printer in a session will be the first printer auto-created in the session, which is either:
    - The first printer added locally to the Windows server in Control Panel > Devices and Printers.
    - The first auto-created printer, if there are no printers added locally to the server.
When setting the session printers value:

- To add printers, type the UNC path of the printer you want to auto-create. After adding the printer, you can apply customized settings for the current session at every logon.

Printer auto-creation event log preference

This setting specifies the events that are logged during the printer auto-creation process. You can choose to log no errors or warnings, only errors, or errors and warnings.

By default, errors and warnings are logged.

An example of a warning is an event in which a printer's native driver could not be installed and the Universal print driver is installed instead. To use the Universal print driver in this scenario, configure the Universal print driver usage setting to Use universal printing only or Use universal printing only if requested driver is unavailable.

Session printers

This setting specifies the network printers to be auto-created in a session.

By default, no printers are specified.

To add printers, type the UNC path of the printer you want to auto-create. After adding the printer, you can apply customized settings for the current session at every logon.

Wait for printers to be created (desktop)

This setting allows or prevents a delay in connecting to a session so that desktop printers can be auto-created.

By default, a connection delay does not occur.
Client Printers policy settings

Dec 13, 2013
The Client Printers section contains policy settings for client printers, including settings to autocreate client printers, retain printer properties, and connect to print servers.

Auto-create client printers

This setting specifies the client printers that are auto-created. This setting overrides default client printer auto-creation settings.

By default, all client printers are auto-created.

This setting takes effect only if the Client printer redirection setting is present and set to Allowed.

When adding this setting to a policy, select an option:
- Auto-create all client printers automatically creates all printers on a user device.
- Auto-create the client’s default printer only automatically creates only the printer selected as the default printer on the user device.
- Auto-create local (non-network) client printers only automatically creates only printers directly connected to the user device through an LPT, COM, USB, TCP/IP, or other local port.
- Do not auto-create client printers turns off autocreation for all client printers when users log on. This causes the Remote Desktop Services (RDS) settings for autocreating client printers to override this setting in lower priority policies.

Auto-create generic universal printer

This setting enables or disables autocreation of the generic Citrix Universal Printer object for sessions where a user device compatible with Universal Printing is in use.

By default, the generic Universal Printer object is not autocreated.

Related policy settings
- Universal print driver usage
- Universal driver preference

Client printer names

This setting selects the naming convention for auto-created client printers.

By default, standard printer names are used.

Select Standard printer names to use printer names which are similar to those created by native Remote Desktop Services, such as “HPLaserJet 4 from clientname in session 3.”

Select Legacy printer names to use old-style client printer names and preserve backward compatibility for users or groups using MetaFrame Presentation Server 3.0 or earlier. An example of a legacy printer name is “Client/clientname#/HPLaserJet 4.” Because this option is less secure, use it only to provide backward compatibility for users or groups using MetaFrame Presentation Server 3.0 or earlier.

Note: This option is provided only for backwards compatibility with legacy versions of XenApp and XenDesktop.

Direct connections to print servers
This setting enables or disables direct connections from the virtual desktop or server hosting applications to a print server for client printers hosted on an accessible network share.

By default, direct connections are enabled.

Enable direct connections if the network print server is not across a WAN from the virtual desktop or server hosting applications. Direct communication results in faster printing if the network print server and the virtual desktop or server hosting applications are on the same LAN.

Disable direct connections if the network is across a WAN or has substantial latency or limited bandwidth. Print jobs are routed through the user device where they are redirected to the network print server. Data sent to the user device is compressed, so less bandwidth is consumed as the data travels across the WAN.

If two network printers have the same name, the printer on the same network as the user device is used.

Printer driver mapping and compatibility

This setting specifies the driver substitution rules for auto-created client printers.

By default, no rules are specified.

When you define driver substitution rules, you can allow or prevent printers to be created with the specified driver. Additionally, you can allow created printers to use only universal print drivers. Driver substitution overrides or maps printer driver names the user device provides, substituting an equivalent driver on the server. This gives server applications access to client printers that have the same drivers as the server, but different driver names.

You can add a driver mapping, edit an existing mapping, override custom settings for a mapping, remove a mapping, or change the order of driver entries in the list. When adding a mapping, enter the client printer driver name and then select the server driver you want to substitute.

Printer properties retention

This setting specifies whether or not to store printer properties and where to store them.

By default, the system determines if printer properties are stored on the user device, if available, or in the user profile.

When adding this setting to a policy, select an option:

- **Saved on the client device only** is for user devices that have a mandatory or roaming profile that is not saved. Choose this option only if all the servers in your farm are running XenApp 5 and above and your users are using Citrix online plug-in versions 9.x, 10.x, 11.x, and 12.x, or Citrix Receiver 3.x.
- **Retained in user profile only** is for user devices constrained by bandwidth (this option reduces network traffic) and logon speed or for users with legacy plug-ins. This option stores printer properties in the user profile on the server and prevents any properties exchange with the user device. Use this option with MetaFrame Presentation Server 3.0 or earlier and MetaFrame Presentation Server Client 8.x or earlier. Note that this is applicable only if a Remote Desktop Services (RDS) roaming profile is used.
- **Held in profile only if not saved on client** allows the system to determine where printer properties are stored. Printer properties are stored either on the user device, if available, or in the user profile. Although this option is the most flexible, it can also slow logon time and use extra bandwidth for system-checking.
- **Do not retain printer properties** prevents storing printer properties.

Retained and restored client printers
This setting enables or disables the retention and re-creation of printers on the user device. By default, client printers are auto-retained and auto-restored.

Retained printers are user-created printers that are created again, or remembered, at the start of the next session. When XenApp recreates a retained printer, it considers all policy settings except the Auto-create client printers setting.

Restored printers are printers fully customized by an administrator, with a saved state that is permanently attached to a client port.
Drivers policy settings

Dec 13, 2013
The Drivers section contains policy settings related to printer drivers.

Automatic installation of in-box printer drivers

This setting enables or disables the automatic installation of printer drivers from the Windows in-box driver set or from driver packages staged on the host using pnputil.exe /a.

By default, these drivers are installed as needed.

Universal driver preference

This setting specifies the order in which Universal printer drivers are used, beginning with the first entry in the list.

By default, the preference order is as follows:
- EMF
- XPS
- PCL5c
- PCL4
- PS

You can add, edit, or remove drivers, and change the order of drivers in the list.

Universal print driver usage

This setting specifies when to use universal printing.

By default, universal printing is used only if the requested driver is unavailable.

Universal printing employs generic printer drivers instead of standard model-specific drivers, potentially simplifying the burden of driver management on host computers. The availability of universal print drivers depends on the capabilities of the user device, host, and print server software. In certain configurations, universal printing might not be available.

When adding this setting to a policy, select an option:
- Use only printer model specific drivers specifies that the client printer uses only the standard model-specific drivers that are auto-created at logon. If the requested driver is unavailable, the client printer cannot be auto-created.
- Use universal printing only specifies that no standard model-specific drivers are used. Only universal print drivers are used to create printers.
- Use universal printing only if requested driver is unavailable uses standard model-specific drivers for printer creation if they are available. If the driver is not available on the server, the client printer is created automatically with the appropriate universal driver.
- Use printer model specific drivers only if universal printing is unavailable uses the universal print driver if it is available. If the driver is not available on the server, the client printer is created automatically with the appropriate model-specific printer driver.
Universal Print Server policy settings

Dec 13, 2013
The Universal Print Server section contains policy settings for handling the Universal Print Server.

Universal Print Server enable

This setting enables or disables the Universal Print Server feature on the virtual desktop or server hosting applications. Apply this policy setting to Organisational Units (OUs) containing the virtual desktop or server hosting applications.

By default, the Universal Print Server is disabled.

When adding this setting to a policy, select one of the following options:
- **Enabled with fallback to Windows native remote printing.** Network printer connections are serviced by the Universal Print Server, if possible. If the Universal Print Server is not available, the Windows Print Provider is used. The Windows Print Provider continues to handle all printers previously created with the Windows Print Provider.
- **Enabled with no fallback to Windows native remote printing.** Network printer connections are serviced by the Universal Print Server exclusively. If the Universal Print Server is unavailable, the network printer connection fails. This setting effectively disables network printing through the Windows Print Provider. Printers previously created with the Windows Print Provider are not created while a policy containing this setting is active.
- **Disabled.** The Universal Print Server feature is disabled. No attempt is made to connect with the Universal Print Server when connecting to a network printer with a UNC name. Connections to remote printers continue to use the Windows native remote printing facility.

Universal Print Server print data stream (CGP) port

This setting specifies the TCP port number used by the Universal Print Server print data stream Common Gateway Protocol (CGP) listener. Apply this policy setting only to OUs containing the print server.

By default, the port number is set to 7229.

Valid port numbers must be in the range of 1 to 65535.

Universal Print Server print stream input bandwidth limit (kpbs)

This setting specifies the upper boundary (in kilobits per second) for the transfer rate of print data delivered from each print job to the Universal Print Server using CGP. Apply this policy setting to OUs containing the virtual desktop or server hosting applications.

By default, the value is 0, which specifies no upper boundary.

Universal Print Server web service (HTTP/SOAP) port

This setting specifies the TCP port number used by the Universal Print Server Web service listener for incoming HTTP/SOAP requests. Ensure that the value specified for this setting is identical for both the OU containing the print server and the OU containing the virtual desktop or server hosting applications.

By default, the port number is set to 8080.

Valid port numbers must be in the range of 0 to 65535.
Universal Printing policy settings

Jan 20, 2014

The Universal Printing section contains policy settings for managing universal printing.

Universal printing EMF processing mode

This setting controls the method of processing the EMF spool file on the Windows user device.

By default, EMF records are spooled directly to the printer.

When adding this setting to a policy, select an option:

- Reprocess EMFs for printer forces the EMF spool file to be reprocessed and sent through the GDI subsystem on the user device. You can use this setting for drivers that require EMF reprocessing but that might not be selected automatically in a session.
- Spool directly to printer, when used with the Citrix Universal print driver, ensures the EMF records are spooled and delivered to the user device for processing. Typically, these EMF spool files are injected directly to the client’s spool queue. For printers and drivers that are compatible with the EMF format, this is the fastest printing method.

Universal printing image compression limit

This setting specifies the maximum quality and the minimum compression level available for images printed with the Citrix Universal print driver.

By default, the image compression limit is set to Best quality (lossless compression).

If No Compression is selected, compression is disabled for EMF printing only.

When adding this setting to a policy, select an option:

- No compression
- Best quality (lossless compression)
- High quality
- Standard quality
- Reduced quality (maximum compression)

When adding this setting to a policy that includes the Universal printing optimization defaults setting, be aware of the following items:

- If the compression level in the Universal printing image compression limit setting is lower than the level defined in the Universal printing optimization defaults setting, images are compressed at the level defined in the Universal printing image compression limits setting.
- If compression is disabled, the Desired image quality and Enable heavyweight compression options of the Universal printing optimization defaults setting have no effect in the policy.

Universal printing optimization defaults

This setting specifies the default values for printing optimization when the universal print driver is created for a session.

- Desired image quality specifies the default image compression limit applied to universal printing. By default, Standard Quality is enabled, meaning that users can only print images using standard or reduced quality compression.
- Enable heavyweight compression enables or disables reducing bandwidth beyond the compression level set by Desired image quality, without losing image quality. By default, heavyweight compression is disabled.
• Image and Font Caching settings specify whether or not to cache images and fonts that appear multiple times in the print stream, ensuring each unique image or font is sent to the printer only once. By default, embedded images and fonts are cached. Note that these settings apply only if the user device supports this behavior.
• Allow non-administrators to modify these settings specifies whether or not users can change the default print optimization settings within a session. By default, users are not allowed to change the default print optimization settings.

Note: All of these options are supported for EMF printing. For XPS printing, only the Desired image quality option is supported.

When adding this setting to a policy that includes the Universal printing image compression limit setting, be aware of the following items:
• If the compression level in the Universal printing image compression limit setting is lower than the level defined in the Universal printing optimization defaults setting, images are compressed at the level defined in the Universal printing image compression limits setting.
• If compression is disabled, the Desired image quality and Enable heavyweight compression options of the Universal printing optimization defaults setting have no effect in the policy.

Universal printing preview preference

This setting specifies whether or not to use the print preview function for auto-created or generic universal printers.

By default, print preview is not used for auto-created or generic universal printers.

When adding this setting to a policy, select an option:
• Do not use print preview for auto-created or generic universal printers
• Use print preview for auto-created printers only
• Use print preview for generic universal printers only
• Use print preview for both auto-created and generic universal printers

Universal printing print quality limit

This setting specifies the maximum dots per inch (dpi) available for generating printed output in a session.

By default, No Limit is enabled, meaning users can select the maximum print quality allowed by the printer to which they connect.

If this setting is configured, it limits the maximum print quality available to users in terms of output resolution. Both the print quality itself and the print quality capabilities of the printer to which the user connects are restricted to the configured setting. For example, if configured to Medium Resolution (600 DPI), users are restricted to printing output with a maximum quality of 600 DPI and the Print Quality setting on the Advanced tab of the Universal Printer dialog box shows resolution settings only up to and including Medium Quality (600 DPI).

When adding this setting to a policy, select an option:
• Draft (150 DPI)
• Low Resolution (300 DPI)
• Medium Resolution (600 DPI)
• High Resolution (1200 DPI)
• No Limit
Security policy settings

Apr 22, 2015
The Security section contains the policy setting for configuring session encryption and encryption of logon data.

SecureICA minimum encryption level

This setting specifies the minimum level at which to encrypt session data sent between the server and a user device.

Important:
For the Virtual Delivery Agent 7.x, this policy setting can be used only to enable the encryption of the logon data with RC5 128-bit encryption. Other settings are provided only for backwards compatibility with legacy versions of XenApp and XenDesktop.

For the VDA 7.x, encryption of session data is set using the basic settings of the VDA's Delivery group. If Enable Secure ICA is selected for the Delivery group, session data is encrypted with RC5 (128 bit) encryption. If Enable Secure ICA is not selected for the Delivery group, session data is encrypted with Basic encryption.

When adding this setting to a policy, select an option:
- Basic encrypts the client connection using a non-RC5 algorithm. It protects the data stream from being read directly, but it can be decrypted. By default, the server uses Basic encryption for client-server traffic.
- RC5 (128 bit) logon only encrypts the logon data with RC5 128-bit encryption and the client connection using Basic encryption.
- RC5 (40 bit) encrypts the client connection with RC5 40-bit encryption.
- RC5 (56 bit) encrypts the client connection with RC5 56-bit encryption.
- RC5 (128 bit) encrypts the client connection with RC5 128-bit encryption.

The settings you specify for client-server encryption can interact with any other encryption settings in your environment and your Windows operating system. If a higher priority encryption level is set on either a server or user device, settings you specify for published resources can be overridden.

You can raise encryption levels to further secure communications and message integrity for certain users. If a policy requires a higher encryption level, Receivers using a lower encryption level are denied connection.

SecureICA does not perform authentication or check data integrity. To provide end-to-end encryption for your site, use SecureICA with SSL/TLS encryption.

SecureICA does not use FIPS-compliant algorithms. If this is an issue, configure the server and Receivers to avoid using SecureICA.
Server Limits policy settings

Dec 13, 2013
The Server Limits section contains policy settings for controlling idle connections.

Server idle timer interval

This setting determines, in milliseconds, how long an uninterrupted user session is maintained if there is no input from the user.

By default, idle connections are not disconnected (Server idle timer interval = 0).
Session Limits policy settings

Dec 13, 2013

The Session Limits section contains policy settings you can use to control how long sessions remain connected before they are forced to log off.

Disconnected session timer

This setting enables or disables a timer to determine how long a disconnected, locked desktop can remain locked before the session is logged off.

By default, disconnected sessions are not logged off.

Disconnected session timer interval

This setting determines how long, in minutes, a disconnected, locked desktop can remain locked before the session is logged off.

By default, the time period is 1440 minutes (24 hours).

Session connection timer

This setting enables or disables a timer to determine the maximum duration of an uninterrupted connection between a user device and a desktop.

By default, this timer is disabled.

Session connection timer interval

This setting determines, in minutes, the maximum duration of an uninterrupted connection between a user device and a desktop.

By default, the maximum duration is 1440 minutes (24 hours).

Session idle timer

This setting enables or disables a timer to determine how long an uninterrupted user device connection to a desktop will be maintained if there is no input from the user.

By default, this timer is enabled.

Session idle timer interval

This setting determines, in minutes, how long an uninterrupted user device connection to a desktop will be maintained if there is no input from the user.

By default, idle connections are maintained for 1440 minutes (24 hours).
Session Reliability policy settings

Dec 13, 2013

The Session Reliability section contains policy settings for managing session reliability connections.

Session reliability connections

This setting allows or prevents sessions to remain open during a loss of network connectivity.

By default, session reliability is allowed.

Session Reliability keeps sessions active and on the user's screen when network connectivity is interrupted. Users continue to see the application they are using until network connectivity resumes.

With Session Reliability, the session remains active on the server. To indicate that connectivity is lost, the user's display freezes and the cursor changes to a spinning hourglass until connectivity resumes on the other side of the tunnel. The user continues to access the display during the interruption and can resume interacting with the application when the network connection is restored. Session Reliability reconnects users without reauthentication prompts.

If you do not want users to be able to reconnect to interrupted sessions without having to reauthenticate, configure the Auto client reconnect authentication setting to require authentication. Users are then prompted to reauthenticate when reconnecting to interrupted sessions.

If you use both Session Reliability and Auto Client Reconnect, the two features work in sequence. Session Reliability closes, or disconnects, the user session after the amount of time you specify in the Session reliability timeout setting. After that, the settings you configure for Auto Client Reconnect take effect, attempting to reconnect the user to the disconnected session.

Session reliability port number

This setting specifies the TCP port number for incoming session reliability connections.

By default, the port number is set to 2598.

Session reliability timeout

This setting specifies the length of time, in seconds, the session reliability proxy waits for a user to reconnect before allowing the session to be disconnected.

By default, this is set to 180 seconds, or three minutes.

Though you can extend the amount of time a session is kept open, this feature is designed to be convenient to the user and it does not prompt the user for reauthentication. As you extend the amount of time a session is kept open, chances increase that a user may get distracted and walk away from the user device, potentially leaving the session accessible to unauthorized users.

If you do not want users to be able to reconnect to interrupted sessions without having to reauthenticate, configure the Auto client reconnect authentication setting to require authentication. Users are then prompted to reauthenticate when reconnecting to interrupted sessions.

If you use both Session Reliability and Auto Client Reconnect, the two features work in sequence. Session Reliability closes,
or disconnects, the user session after the amount of time you specify in the Session reliability timeout setting. After that, the settings you configure for Auto Client Reconnect take effect, attempting to reconnect the user to the disconnected session.
Time Zone Control policy settings

Dec 13, 2013
The Time Zone Control section contains policy settings related to using local time in sessions.

Estimate local time for legacy clients

This setting enables or disables estimating the local time zone of user devices that send inaccurate time zone information to the server.

By default, the server estimates the local time zone when necessary.

Use local time of client

This setting determines the time zone setting of the user session. This can be either the time zone of the user session or the time zone of the user device.

By default, the time zone of the user session is used.

For this setting to take effect, enable the Allow time zone redirection setting in the Remote Desktop Session Host node of the Group Policy Management Editor (User Configuration > Policies > Administrative Templates > Windows Components > Remote Desktop Services > Remote Desktop Session Host > Device and Resource Redirection). For more information about time zone redirection, refer to the Citrix Knowledge Center.
TWAIN Devices policy settings

Dec 13, 2013

The TWAIN devices section contains policy settings related to mapping client TWAIN devices, such as digital cameras or scanners, and optimizing image transfers from server to client.

Client TWAIN device redirection

This setting allows or prevents users from accessing TWAIN devices on the user device from image processing applications hosted on servers. By default, TWAIN device redirection is allowed.

Related policy settings

- TWAIN compression level
- TWAIN device redirection bandwidth limit
- TWAIN device redirection bandwidth limit percent

TWAIN compression level

This setting specifies the level of compression of image transfers from client to server. Use Low for best image quality, Medium for good image quality, or High for low image quality. By default, medium compression is applied.
USB Devices policy settings

Jan 20, 2014

The USB devices section contains policy settings for managing file redirection for USB devices.

Client USB device redirection

This setting allows or prevents redirection of USB devices to and from the user device. By default, USB devices are not redirected.

Client USB device redirection rules

This setting specifies redirection rules for USB devices. By default, no rules are specified.

When a user plugs in a USB device, the host device checks it against each policy rule in turn until a match is found. The first match for any device is considered definitive. If the first match is an Allow rule, the device is remoted to the virtual desktop. If the first match is a Deny rule, the device is available only to the local desktop. If no match is found, default rules are used.

Policy rules take the format {Allow:|Deny:} followed by a set of tag= value expressions separated by whitespace. The following tags are supported:

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VID</td>
<td>Vendor ID from the device descriptor</td>
</tr>
<tr>
<td>PID</td>
<td>Product ID from the device descriptor</td>
</tr>
<tr>
<td>REL</td>
<td>Release ID from the device descriptor</td>
</tr>
<tr>
<td>Class</td>
<td>Class from either the device descriptor or an interface descriptor</td>
</tr>
<tr>
<td>SubClass</td>
<td>Subclass from either the device descriptor or an interface descriptor</td>
</tr>
<tr>
<td>Prot</td>
<td>Protocol from either the device descriptor or an interface descriptor</td>
</tr>
</tbody>
</table>

When creating new policy rules, be aware of the following:

- Rules are case-insensitive.
- Rules may have an optional comment at the end, introduced by #.
- Blank and pure comment lines are ignored.
- Tags must use the matching operator =. For example, VID=1230.
- Each rule must start on a new line or form part of a semicolon-separated list.
- Refer to the USB class codes available from the USB Implementers Forum, Inc. Web site.

Examples of administrator-defined USB policy rules

Allow: VID=1230 PID=0007 # ANOther Industries, ANOther Flash Drive

Deny: Class=08 subclass=05 # Mass Storage
To create a rule that denies all USB devices, use “DENY:” with no other tags.

Client USB Plug and Play device redirection

This setting allows or prevents plug-and-play devices such as cameras or point-of-sale (POS) devices to be used in a client session.

By default, plug-and-play device redirection is allowed. When set to Allowed, all plug-and-play devices for a specific user or group are redirected. When set to Prohibited, no devices are redirected.
Visual Display policy settings

Dec 13, 2013
The Visual Display section contains policy settings for controlling the quality of images sent from virtual desktops to the user device.

Target frame rate

This setting specifies the maximum number of frames per second sent to the user device from the virtual desktop.

By default, the maximum is 30 frames per second.

Setting a high number of frames per second (for example, 30) improves the user experience, but requires more bandwidth. Decreasing the number of frames per second (for example, 10) maximizes server scalability at the expense of user experience.

Visual quality

This setting specifies the desired visual quality for images displayed on the user device.

By default, this is set to Medium.

To specify the quality of images, choose one of the following options:

- Low
- Medium
- High
- Build to lossless
- Always lossless

In cases where preserving image data is vital, for example, when displaying X-ray images where no loss of quality is acceptable, select Always lossless to ensure lossy data is never sent to the user device.

Selecting Build to lossless sends lossy images to the user device during periods of high network activity and lossless images after network activity reduces.

Note: If the Legacy graphics mode setting is enabled for a policy, the Visual quality setting has no effect in that policy.
Moving Images policy settings

Dec 13, 2013
The Moving Images section contains settings that enable you to remove or alter compression for dynamic images.

Minimum image quality

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled. This setting specifies the minimum acceptable image quality for Adaptive Display. The less compression used, the higher the quality of images displayed. Choose from Ultra High, Very High, High, Normal, or Low compression.

By default, this is set to Normal.

Moving image compression

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled. This setting specifies whether or not Adaptive Display is enabled. Adaptive Display automatically adjusts the image quality of videos and transitional slides in slide shows based on available bandwidth. With Adaptive Display enabled, users should see smooth-running presentations with no reduction in quality.

By default, Adaptive Display is enabled.

Progressive compression level

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled. This setting provides a less detailed but faster initial display of images.

By default, no progressive compression is applied.

The more detailed image, defined by the normal lossy compression setting, appears when it becomes available. Use Very High or Ultra High compression for improved viewing of bandwidth-intensive graphics such as photographs.

For progressive compression to be effective, its compression level must be higher than the Lossy compression level setting. Note: The increased level of compression associated with progressive compression also enhances the interactivity of dynamic images over client connections. The quality of a dynamic image, such as a rotating three-dimensional model, is temporarily decreased until the image stops moving, at which time the normal lossy compression setting is applied.

Related policy settings
- Progressive compression threshold value
- Progressive heavyweight compression

Progressive compression threshold value

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled. This setting represents the maximum bandwidth in kilobits per second for a connection to which progressive compression is applied. This is applied only to client connections under this bandwidth.

By default, the threshold value is 2147483647 kilobits per second.

Related policy settings
- Progressive compression threshold value
- Progressive heavyweight compression
Target minimum frame rate

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled. This setting specifies the minimum frame rate per second the system attempts to maintain, for dynamic images, under low bandwidth conditions.

By default, this is set to 10fps.
Still Images policy settings

Dec 13, 2013
The Still Images section contains settings that enable you to remove or alter compression for static images.

Extra color compression

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled.
This setting enables or disables the use of extra color compression on images delivered over client connections that are limited in bandwidth, improving responsiveness by reducing the quality of displayed images.

By default, extra color compression is disabled.

When enabled, extra color compression is applied only when the client connection bandwidth is below the Extra color compression threshold value. When the client connection bandwidth is above the threshold value or Disabled is selected, extra color compression is not applied.

Extra color compression threshold

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled.
This setting represents the maximum bandwidth in kilobits per second for a connection below which extra color compression is applied. If the client connection bandwidth drops below the set value, extra color compression, if enabled, is applied.

By default, the threshold value is 8192 kilobits per second.

Heavyweight compression

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled.
This setting enables or disables reducing bandwidth beyond progressive compression without losing image quality by using a more advanced, but more CPU-intensive, graphical algorithm.

By default, heavyweight compression is disabled.

If enabled, heavyweight compression applies to all lossy compression settings. It is supported on Citrix Receiver but has no effect on other plug-ins.

Related policy settings

- Progressive compression level
- Progressive compression threshold value

Lossy compression level

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled.
This setting controls the degree of lossy compression used on images delivered over client connections that are limited in bandwidth. In such cases, displaying images without compression can be slow.

By default, medium compression is selected.

For improved responsiveness with bandwidth-intensive images, use high compression. Where preserving image data is vital, for example, when displaying X-ray images where no loss of quality is acceptable, you may not want to use lossy
Related policy settings

- Lossy compression threshold value

Lossy compression threshold value

Note: In XenDesktop 7.0, this policy setting applies only when the Legacy graphics mode policy setting is enabled. This setting represents the maximum bandwidth in kilobits per second for a connection to which lossy compression is applied.

By default, the threshold value is 2147483647 kilobits per second.

Adding the Lossy compression level setting to a policy and including no specified threshold can improve the display speed of high-detail bitmaps, such as photographs, over a LAN.

Related policy settings

- Lossy compression level
WebSockets policy settings

Dec 13, 2013
The WebSockets section contains policy settings for accessing virtual desktops and hosted applications with Receiver for HTML5. The WebSockets feature increases security and reduces overhead by conducting two-way communication between browser-based applications and servers without opening multiple HTTP connections.

WebSockets connections
This setting allows or prohibits WebSockets connections.

By default, WebSocket connections are prohibited.

WebSockets port number
This setting identifies the port for incoming WebSocket connections.

By default, the value is 8008.

WebSockets trusted origin server list
This setting provides a comma-separated list of trusted origin servers, usually Receiver for Web, expressed as URLs. Only WebSockets connections originating from one of these addresses is accepted by the server.

By default, the wildcard * is used to trust all Receiver for Web URLs.

If you choose to type an address in the list, use this syntax:

<protocol>://<Fully qualified domain name of host>[port]

The protocol should be HTTP or HTTPS. If the port is not specified, port 80 is used for HTTP and port 443 is used for HTTPS.

The wildcard * can be used within the URL, except as part of an IP address (10.105.*.*).
Load Management policy settings

Dec 13, 2013
The Load Management section contains policy settings for enabling and configuring load management between servers delivering Windows Server OS machines.

Concurrent logon tolerance
This setting specifies the maximum number of concurrent logons a server can accept.
By default, this is set to 2.

CPU usage
This setting specifies the level of CPU usage, as a percentage, at which the server reports a full load. When enabled, the default value at which the server reports a full load is 90%.
By default, this setting is disabled and CPU usage is excluded from load calculations.

CPU usage excluded process priority
This setting specifies the priority level at which a process' CPU usage is excluded from the CPU Usage load index.
By default, this is set to Below Normal or Low.

Disk usage
This setting specifies the disk queue length at which the server reports a 75% full load. When enabled, the default value for disk queue length is 8.
By default, this setting is disabled and disk usage is excluded from load calculations.

Maximum number of sessions
This setting specifies the maximum number of sessions a server can host. When enabled, the default setting for maximum number of sessions a server can host is 250.
By default, this setting is enabled.

Memory usage
This setting specifies the level of memory usage, as a percentage, at which the server reports a full load. When enabled, the default value at which the server reports a full load is 90%.
By default, this setting is disabled and memory usage is excluded from load calculations.

Memory usage base load
This setting specifies an approximation of the base operating system's memory usage and defines, in MB, the memory usage below which a server is considered to have zero load.
By default, this is set to 768MB.
Microsoft Application Virtualization (App-V) policy settings

Dec 13, 2013
The Microsoft Application Virtualization (App-V) section, contains policy settings for configuring App-V servers for use within your environment.

App-V servers

This setting specifies a list of App-V servers from which App-V applications are published within your environment.

Important: You must configure this setting using Active Directory's Group Policy Editor.
Profile Management policy settings

Mar 27, 2013

The Profile Management section contains policy settings for enabling Profile management and specifying which groups to include in and exclude from Profile management processing.

Other information, such as the names of the equivalent .ini file settings and which version of Profile management is required for any particular policy setting, is available in Profile Management ADM File Reference.
Advanced policy settings

Dec 13, 2013

The Advanced settings section contains policy settings relating to the advanced configuration of Profile management.

Disable automatic configuration

This setting enables Profile management to examine your environment; for example, to check for the presence of personal vDisks, and configure Group Policy accordingly. Only Profile management policies in the Not Configured state are adjusted, so any customizations made previously are preserved. This feature speeds up deployment and simplifies optimization. No configuration of the feature is necessary, but you can disable automatic configuration when upgrading (to retain settings from earlier versions) or when troubleshooting. Automatic configuration does not work in XenApp or other environments.

By default, automatic configuration is allowed.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, automatic configuration is turned on so Profile management settings might change if your environment changes.

Log off user if a problem is encountered

This setting enables Profile management to log a user off if a problem is encountered; for example, if the user store is unavailable. When enabled, an error message is displayed to the user before they are logged off. When disabled, users are given a temporary profile.

By default, this setting is disabled and users are given a temporary profile if a problem is encountered.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, a temporary profile is provided.

Number of retries when accessing locked files

This setting specifies the number of attempts Profile management makes to access locked files.

By default, this is set to five retries.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, the default value is used.

Process Internet cookie files on logoff

This setting enables Profile management to process index.dat on logoff to remove Internet cookies left in the file system after sustained browsing that can lead to profile bloat. Enabling this setting increases logoff times, so only enable it if you experience this issue.

By default, this setting is disabled and Profile management does not process index.dat on logoff.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no processing of Index.dat takes place.
Basic policy settings

Feb 17, 2014

The Basic settings section contains policy settings relating to the basic configuration of Profile management.

Active write back

This setting enables modified files and folders (but not registry settings) to be synchronized to the user store during a session, before logoff.

By default, synchronization to the user store during a session is disabled.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, it is enabled.

Enable Profile management

This setting enables Profile management to process logons and logoffs.

By default, this setting is disabled to facilitate deployment.

Important: Citrix recommends enabling Profile management only after carrying out all other setup tasks and testing how Citrix user profiles perform in your environment.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, Profile management does not process Windows user profiles in any way.

Excluded groups

This setting specifies which computer local groups and domain groups (local, global, and universal) are excluded from Profile management processing.

When enabled, Profile management does not process members of the specified user groups.

By default, this setting is disabled and members of all user groups are processed.

Specify domain groups in the form <DOMAIN NAME>\<GROUP NAME>.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, members of all user groups are processed.

Offline profile support

This setting enables offline profile support, allowing profiles to synchronize with the user store at the earliest opportunity after a network disconnection.

By default, support for offline profiles is disabled.

This setting is applicable to laptop or mobile users who roam. When a network disconnection occurs, profiles remain intact on the laptop or device even after rebooting or hibernating. As mobile users work, their profiles are updated locally and are
synchronized with the user store when the network connection is re-established.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, support for offline profiles is disabled.

Path to user store

This setting specifies the path to the directory (user store) in which user settings, such as registry settings and synchronized files, are saved.

By default, the Windows directory on the home drive is used.

If this setting is disabled, user settings are saved in the Windows subdirectory of the home directory.

The path can be:
- **A relative path.** This must be relative to the home directory, typically configured as the #homeDirectory# attribute for a user in Active Directory.
- **An absolute UNC path.** This typically specifies a server share or a DFS namespace.
- **Disabled or unconfigured.** In this case, a value of #homeDirectory#\Windows is assumed.

Use the following types of variables when configuring this policy setting:
- System environment variables enclosed in percent signs (for example, %ProfVer%). Note that system environment variables generally require additional setup.
- Attributes of the Active Directory user object enclosed in hashes (for example, #sAMAccountName#).
- Profile management variables. For more information, see the Profile Management section in eDocs.

You can also use the %username% and %userdomain% user environment variables and create custom attributes to fully define organizational variables such as location or users. Attributes are case-sensitive.

Examples:
- `\server\share\#sAMAccountName#` stores the user settings to the UNC path `\server\share\JohnSmith` (if #sAMAccountName# resolves to JohnSmith for the current user)
- `\server\profiles$\%USERNAME%.%USERDOMAIN%\CTX_PROFILEVER!!CTX_OSBITNESS!` might expand to `\server\profiles$\JohnSmith.DOMAINCONTROLLER1\v2x64`

Important: Whichever attributes or variables you use, check that this setting expands to the folder one level higher than the folder containing NTUSER.DAT. For example, if this file is contained in `\server\profiles$\JohnSmith.Finance\v2x64\UPM_Profile`, set the path to the user store as `\server\profiles$\JohnSmith.Finance\v2x64`, not the \UPM_Profile subfolder.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, the Windows directory on the home drive is used.

Process logons of local administrators

This setting specifies whether or not logons of members of the BUILTIN\Administrators group are processed. This allows domain users with local administrator rights, typically users with assigned virtual desktops, to bypass processing, log on, and troubleshoot a desktop experiencing problems with Profile management.

If this setting is disabled or not configured on server operating systems, Profile management assumes that logons by domain users, but not local administrators, must be processed. On desktop operating systems, local administrator logons
are processed.

By default this setting is disabled, and local administrator logons are not processed.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, local administrator logons are not processed.

Processed groups

This setting specifies which computer local groups and domain groups (local, global, and universal) are included in Profile management processing.

When enabled, Profile management processes only members of the specified user groups.

By default, this setting is disabled and members of all user groups are processed.

Specify domain groups in the form <DOMAIN NAME>\<GROUP NAME>.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, members of all user groups are processed.
Cross-Platform policy settings

Dec 13, 2013
The Cross-Platform section contains policy settings relating to configuring the Profile management cross-platform settings feature.

Cross-platform settings user groups
This setting specifies the Windows user groups whose profiles are processed when the cross-platform settings feature is enabled.

By default, this setting is disabled and all user groups specified in the Processed Group policy setting are processed.
If this setting is not configured here, the value from the .ini file is used.
If this setting is not configured here or in the .ini file, all user groups are processed.

Enable cross-platform settings
This setting enables or disables the cross-platform settings feature, that allows you to migrate users' profiles and roam them when a user connects to the same application running on multiple operating systems.

By default the cross-platform settings feature is disabled.
If this setting is not configured here, the value from the .ini file is used.
If this setting is not configured here or in the .ini file, no cross-platform settings are applied.

Path to cross-platform definitions
This setting specifies the network location, as a UNC path, of the definition files copied from the download package.

Note: Users must have read access, and administrators write access, to this location and it must be either a Server Message Block (SMB) or Common Internet File System (CIFS) file share.
By default, no path is specified.

If this setting is not configured here, the value from the .ini file is used.
If this setting is not configured here or in the .ini file, no cross-platform settings are applied.

Path to cross-platform settings store
This setting specifies the path to the cross-settings store, the folder in which users' cross-platform settings are saved. This path can be either a UNC path or a path relative to the home directory.

Note: Users must have write access to the cross-settings store.
By default, this setting is disabled and the path Windows\PM_CP is used.

If this setting is not configured here, the value from the .ini file is used.
If this setting is not configured here or in the .ini file, the default value is used.

Source for creating cross-platform settings
This setting specifies a platform as the
— base platform
if this setting is enabled for that platform's OU. Data from the base platform's profiles is migrated to the cross-platform settings store.

Each platform's own set of profiles are stored in a separate OU. This means you must decide which platform's profile data to use to seed the cross-platform settings store. This is referred to as the
— base platform
.

When enabled, Profile management migrates the data from the single-platform profile to the store if the cross-platform settings store contains a definition file with no data, or if the cached data in a single-platform profile is newer than the definition's data in the store.

Important: If this setting is enabled in multiple OUs, or multiple user or machine objects, the platform that the first user logs on to becomes the base profile.

By default, this setting is disabled and Profile management does not migrate the data from the single-platform profile to the store.
File System policy settings

Mar 27, 2013
The File System section contains policy settings for configuring which files and directories in a users profile are synchronized between the system on which the profile is installed and the user store.
The Exclusions section contains policy settings for configuring which files and directories in a user's profile are excluded from the synchronization process.

### Exclusion list - directories

This setting specifies a list of folders in the user profile that are ignored during synchronization.

Specify folder names as paths relative to the user profile (%USERPROFILE%).

By default, this setting is disabled and all folders in the user profile are synchronized.

**Examples:**
- Desktop ignores the Desktop folder in the user profile

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, all folders in the user profile are synchronized.

### Exclusion list - files

This setting specifies a list of files in the user profile that are ignored during synchronization.

Specify file names as paths relative to the user profile (%USERPROFILE%). Note that wildcards are allowed and are applied recursively.

**Examples:**
- Desktop\Desktop.ini ignores the file Desktop.ini in the Desktop folder

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, all files in the user profile are synchronized.
Synchronization policy settings

Dec 13, 2013

The Synchronization section contains policy settings for specifying which files and folders in a users profile are synchronized between the system on which the profile is installed and the user store.

Directories to synchronize

This setting specifies any folders you want Profile management to include in the synchronization process that are located either outside the user profile or in excluded folders. By default, Profile management synchronizes each user's entire profile between the system it is installed on and the user store. It is not necessary to include subfolders of the user profile by adding them to this list.

Paths on this list must be relative to the user profile.

Examples:
- Desktop\exclude\include ensures that the subfolder called include is synchronized even if the folder called Desktop\exclude is not

By default, this setting is disabled and no folders are specified.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, only non-excluded folders in the user profile are synchronized.

Files to synchronize

This setting specifies any files you want Profile management to include in the synchronization process that are located either outside the user profile or in excluded folders. By default, Profile management synchronizes each user's entire profile between the system it is installed on and the user store. It is not necessary to include files in the user profile by adding them to this list.

Paths on this list must be relative to the user profile. Relative paths are interpreted as being relative to the user profile. Wildcards can be used but are allowed only for file names. Wildcards cannot be nested and are applied recursively.

Examples:
- AppData\Local\Microsoft\Office\Access.qat specifies a file below a folder that is excluded in the default configuration
- AppData\Local\MyApp\.cfg specifies all files with the extension .cfg in the profile folder AppData\Local\MyApp and its subfolders

By default, this setting is disabled and no files are specified.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, only non-excluded files in the user profile are synchronized.

Folders to mirror

This setting specifies which folders relative to a users' profile root folder to mirror. Configuring this policy setting can help solve issues involving any transactional folder (also known as a referential folder), that is a folder containing interdependent files, where one file references others.
Mirroring folders allows Profile management to process a transactional folder and its contents as a single entity, avoiding profile bloat. Be aware that, in these situations the "last write wins" so files in mirrored folders that have been modified in more than one session will be overwritten by the last update, resulting in loss of profile changes.

For example, you can mirror the Internet Explorer cookies folder so that Index.dat is synchronized with the cookies that it indexes.

If a user has two Internet Explorer sessions, each on a different server, and they visit different sites in each session, cookies from each site are added to the appropriate server. When the user logs off from the first session (or in the middle of a session, if the active write back feature is configured), the cookies from the second session should replace those from the first session. However, instead they are merged, and the references to the cookies in Index.dat become out of date. Further browsing in new sessions results in repeated merging and a bloated cookie folder.

Mirroring the cookie folder solves the issue by overwriting the cookies with those from the last session each time the user logs off so Index.dat stays up to date.

By default, this setting is disabled and no folders are mirrored.

If this setting is not configured here, the value from the .ini file is used.

If this policy is not configured here or in the .ini file, no folders are mirrored.
Folder Redirection policy settings

The Folder Redirection section contains policy settings for specifying whether to redirect folders that commonly appear in profiles to a shared network location.

Grant administrator access

This setting enables an administrator to access the contents of a users' redirected folders.

By default, this setting is disabled and users are granted exclusive access to the contents of their redirected folders.

Include domain name

This setting enables the inclusion of the %userdomain% environment variable as part of the UNC path specified for redirected folders.

By default, this setting is disabled and the %userdomain% environment variable is not included as part of the UNC path specified for redirected folders.
AppData(Roaming) policy settings

Dec 13, 2013
The AppData(Roaming) section contains policy settings for specifying whether to redirect the contents the AppData(Roaming) folder to a shared network location.

AppData(Roaming) path
This setting specifies the network location to which the contents of the AppData(Roaming) folder are redirected.

By default, this setting is disabled and no location is specified.

If this setting is not configured here, Profile management does not redirect the specified folder.

Redirection settings for AppData(Roaming)
This setting specifies how to redirect the contents of the AppData(Roaming) folder.

By default, contents are redirected to a UNC path.

If this setting is not configured here, Profile management does not redirect the specified folder.
Contacts policy settings

Dec 13, 2013
The Contacts section contains policy settings for specifying whether to redirect the contents the Contacts folder to a shared network location.

Contacts path
This setting specifies the network location to which the contents of the Contacts folder are redirected.

By default, this setting is disabled and no location is specified.

If this setting is not configured here, Profile management does not redirect the specified folder.

Redirection settings for Contacts
This setting specifies how to redirect the contents of the Contacts folder.

By default, contents are redirected to a UNC path.

If this setting is not configured here, Profile management does not redirect the specified folder.
Desktop policy settings

Dec 13, 2013
The Desktop section contains policy settings for specifying whether to redirect the contents the Desktop folder to a shared network location.

Desktop path

This setting specifies the network location to which the contents of the Contacts folder are redirected.

By default, this setting is disabled and no location is specified.

If this setting is not configured here, Profile management does not redirect the specified folder.

Redirection settings for Desktop

This setting specifies how to redirect the contents of the Desktop folder.

By default, contents are redirected to a UNC path.

If this setting is not configured here, Profile management does not redirect the specified folder.
Documents policy settings

Dec 13, 2013

The Documents section contains policy settings for specifying whether to redirect the contents of the Documents folder to a shared network location.

Documents path

This setting specifies the network location to which files in the Documents folder are redirected.

By default, this setting is disabled and no location is specified.

If this setting is not configured here, Profile management does not redirect the specified folder.

Redirection settings for Documents

This setting specifies how to redirect the contents of the Documents folder.

By default, contents are redirected to a UNC path.

To control how to redirect the contents of the Documents folder, choose one of the following options:

- Redirect to the following UNC path. Redirects content to the UNC path specified in the Documents path policy setting.
- Redirect to the users home directory. Redirects content to the users home directory, typically configured as the #homeDirectory# attribute for a user in Active Directory.

If this setting is not configured here, Profile management does not redirect the specified folder.
Downloads policy settings

Jan 20, 2014
The Downloads section contains policy settings for specifying whether to redirect the contents the Downloads folder to a shared network location.

Downloads path
This setting specifies the network location to which files in the Downloads folder are redirected.

By default, this setting is disabled and no location is specified.

If this setting is not configured here, Profile management does not redirect the specified folder.

Redirection settings for Downloads
This setting specifies how to redirect the contents of the Downloads folder.

By default, contents are redirected to a UNC path.

If this setting is not configured here, Profile management does not redirect the specified folder.
Favorites policy settings

Dec 13, 2013
The Favorites section contains policy settings for specifying whether to redirect the contents of the Favorites folder to a shared network location.

Favorites path
This setting specifies the network location to which the contents of the Favorites folder are redirected.
By default, this setting is disabled and no location is specified.
If this setting is not configured here, Profile management does not redirect the specified folder.

Redirection settings for Favorites
This setting specifies how to redirect the contents of the Favorites folder.
By default, contents are redirected to a UNC path.
If this setting is not configured here, Profile management does not redirect the specified folder.
Links policy settings

Dec 13, 2013
The Links section contains policy settings for specifying whether to redirect the contents the Links folder to a shared network location.

Links path
This setting specifies the network location to which the contents of the Links folder are redirected.

By default, this setting is disabled and no location is specified.

If this setting is not configured here, Profile management does not redirect the specified folder.

Redirection settings for Links
This setting specifies how to redirect the contents of the Links folder.

By default, contents are redirected to a UNC path.

If this setting is not configured here, Profile management does not redirect the specified folder.
Music policy settings

Dec 13, 2013
The Music section contains policy settings for specifying whether to redirect the contents the Music folder to a shared network location.

Music path
This setting specifies the network location to which the contents of the Music folder are redirected.

By default, this setting is disabled and no location is specified.

If this setting is not configured here, Profile management does not redirect the specified folder.

Redirection settings for Music
This setting specifies how to redirect the contents of the Music folder.

By default, contents are redirected to a UNC path.

To control how to redirect the contents of the Music folder, choose one of the following options:

- Redirect to the following UNC path. Redirects content to the UNC path specified in the Music path policy setting.
- Redirect relative to Documents folder. Redirects content to a folder relative to the Documents folder.

If this setting is not configured here, Profile management does not redirect the specified folder.
Pictures policy settings

Dec 13, 2013
The Pictures section contains policy settings for specifying whether to redirect the contents the Pictures folder to a shared network location.

Pictures path
This setting specifies the network location to which the contents of the Pictures folder are redirected.

By default, this setting is disabled and no location is specified.

If this setting is not configured here, Profile management does not redirect the specified folder.

Redirection settings for Pictures
This setting specifies how to redirect the contents of the Pictures folder.

By default, contents are redirected to a UNC path.

To control how to redirect the contents of the Pictures folder, choose one of the following options:

- Redirect to the following UNC path. Redirects content to the UNC path specified in the Pictures path policy setting.
- Redirect relative to Documents folder. Redirects content to a folder relative to the Documents folder.

If this setting is not configured here, Profile management does not redirect the specified folder.
Saved Games policy settings

Dec 13, 2013
The Saved Games section contains policy settings for specifying whether to redirect the contents the Saved Games folder to a shared network location.

Redirection settings for Saved Games
This setting specifies how to redirect the contents of the Saved Games folder.

By default, contents are redirected to a UNC path.

If this setting is not configured here, Profile management does not redirect the specified folder.

Saved Games path
This setting specifies the network location to which the contents of the Saved Games folder are redirected.

By default, this setting is disabled and no location is specified.

If this setting is not configured here, Profile management does not redirect the specified folder.
Searches policy settings

Dec 13, 2013
The Searches section contains policy settings for specifying whether to redirect the contents the Searches folder to a shared network location.

Redirection settings for Searches
This setting specifies how to redirect the contents of the Searches folder.
By default, contents are redirected to a UNC path.
If this setting is not configured here, Profile management does not redirect the specified folder.

Searches path
This setting specifies the network location to which the contents of the Searches folder are redirected.
By default, this setting is disabled and no location is specified.
If this setting is not configured here, Profile management does not redirect the specified folder.
Start Menu policy settings

Dec 13, 2013
The Start Menu section contains policy settings for specifying whether to redirect the contents the Start Menu folder to a shared network location.

Redirection settings for Start Menu
This setting specifies how to redirect the contents of the Start Menu folder.
By default, contents are redirected to a UNC path.
If this setting is not configured here, Profile management does not redirect the specified folder.

Start Menu path
This setting specifies the network location to which the contents of the Start Menu folder are redirected.
By default, this setting is disabled and no location is specified.
If this setting is not configured here, Profile management does not redirect the specified folder.
Video policy settings

Dec 13, 2013
The Video section contains policy settings for specifying whether to redirect the contents the Video folder to a shared network location.

Redirection settings for Video

This setting specifies how to redirect the contents of the Video folder.

By default, contents are redirected to a UNC path.

To control how to redirect the contents of the Video folder, choose one of the following options:
- Redirect to the following UNC path. Redirects content to the UNC path specified in the Video path policy setting.
- Redirect relative to Documents folder. Redirects content to a folder relative to the Documents folder.

If this setting is not configured here, Profile management does not redirect the specified folder.

Video path

This setting specifies the network location to which the contents of the Video folder are redirected.

By default, this setting is disabled and no location is specified.

If this setting is not configured here, Profile management does not redirect the specified folder.
The Log section contains policy settings for configuring Profile management logging.

This setting enables or disables verbose logging of actions performed in Active Directory.
By default, this setting is disabled.
When enabling this setting, make sure the Enable logging setting is also enabled.
If this setting is not configured here, the value from the .ini file is used.
If this setting is not configured here or in the .ini file, errors and general information are logged.

This setting enables or disables verbose logging of common information.
By default, this setting is disabled.
When enabling this setting, make sure the Enable logging setting is also enabled.
If this setting is not configured here, the value from the .ini file is used.
If this setting is not configured here or in the .ini file, errors and general information are logged.

This setting enables or disables verbose logging of common warnings.
By default, this setting is disabled.
When enabling this setting, make sure the Enable logging setting is also enabled.
If this setting is not configured here, the value from the .ini file is used.
If this setting is not configured here or in the .ini file, errors and general information are logged.

This setting enables or disables Profile management logging in debug (verbose logging) mode. In debug mode, extensive status information is logged in the log files located in "%SystemRoot%\System32\Logfiles\UserProfileManager".
By default, this setting is disabled and only errors are logged.
Citrix recommends enabling this setting only if you are troubleshooting Profile management.
If this setting is not configured here, the value from the .ini file is used.
If this setting is not configured here or in the .ini file, only errors are logged.
This setting enables or disables verbose logging of actioned performed in the file system.

By default, this setting is disabled.

When enabling this setting, make sure the Enable logging setting is also enabled.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, errors and general information are logged.

This setting enables or disables verbose logging of file systems notifications.

By default, this setting is disabled.

When enabling this setting, make sure the Enable logging setting is also enabled.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, errors and general information are logged.

This setting enables or disables verbose logging of user logoffs.

By default, this setting is disabled.

When enabling this setting, make sure the Enable logging setting is also enabled.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, errors and general information are logged.

This setting enables or disables verbose logging of user logons.

By default, this setting is disabled.

When enabling this setting, make sure the Enable logging setting is also enabled.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, errors and general information are logged.

This setting specifies the maximum permitted size for the Profile management log file, in bytes.

By default, this is set to 1048576 bytes (1MB).

Citrix recommends increasing the size of this file to 5 MB or more, if you have sufficient disk space. If the log file grows beyond the maximum size, an existing backup of the file (.bak) is deleted, the log file is renamed to .bak, and a new log file is
The log file is created in %SystemRoot%\System32\Logfiles\UserProfileManager.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, the default value is used.

This setting specifies an alternative path to which to save the Profile management log file.

By default, this setting is disabled and log files are saved in the default location; %SystemRoot%\System32\Logfiles\UserProfileManager.

The path can point to a local drive or a remote, network-based drive (a UNC path). Remote paths can be useful in large, distributed environments but they may create significant network traffic, which may be inappropriate for log files. For provisioned, virtual machines with a persistent hard drive, set a local path to that drive. This ensures log files are preserved when the machine restarts. For virtual machines without a persistent hard drive, setting a UNC path allows you to retain the log files but the system account for the machines must have write access to the UNC share. Use a local path for any laptops managed by the offline profiles feature.

If a UNC path is used for log files, Citrix recommends that an appropriate access control list is applied to the log file folder to ensure that only authorized user or computer accounts can access the stored files.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, %SystemRoot%\System32\Logfiles\UserProfileManager is used.

This setting enables or disables verbose logging of personalized user information.

By default, this setting is disabled.

When enabling this setting, make sure the Enable logging setting is also enabled.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, errors and general information are logged.

This setting enables or disables verbose logging of policy values when a user logs on and off.

By default, this setting is disabled.

When enabling this setting, make sure the Enable logging setting is also enabled.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, errors and general information are logged.
This setting enables or disables verbose logging of actions performed in the registry.

By default, this setting is disabled.

When enabling this setting, make sure the Enable logging setting is also enabled.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, errors and general information are logged.

This setting enables or disables verbose logging of any differences in the registry when a user logs off.

By default, this setting is disabled.

When enabling this setting, make sure the Enable logging setting is also enabled.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, errors and general information are logged.
Profile handling policy settings

Dec 13, 2013
The Profile handling section contains policy settings for configuring how Profile management handles user profiles.

This setting specifies an optional extension to the delay, in minutes, before Profile management deletes locally cached profiles at logoff.

A value of 0 deletes the profiles immediately, at the end of the logoff process. Profile management checks for logoffs every minute, so a value of 60 ensures that profiles are deleted between one and two minutes after users have logged off (depending on when the last check took place). Extending the delay is useful if you know that a process keeps files or the user registry hive open during logoff. With large profiles, this can also speed up logoff.

By default, this is set to 0 and Profile management deletes locally cached profiles immediately.

When enabling this setting, ensure the Delete locally cached profiles on logoff is also enabled.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, profiles are deleted immediately.

This setting specifies whether locally cached profiles are deleted after a user logs off.

When this setting is enabled, a user's local profile cache is deleted after they have logged off. Citrix recommends enabling this setting for terminal servers.

By default, this setting is disabled and a user's local profile cache is retained after they log off.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, cached profiles are not deleted.

This setting configures how Profile management behaves if a user profile exists both in the user store and as a local Windows user profile (not a Citrix user profile).

By default, Profile management uses the local Windows profile, but does not change it in any way.

To control how Profile management behaves, choose one of the following options:

- Use local profile. Profile management uses the local profile, but does not change it in any way.
- Delete local profile. Profile management deletes the local Windows user profile, and then imports the Citrix user profile from the user store.
- Rename local profile. Profile management renames the local Windows user profile (for backup purposes) and then imports the Citrix user profile from the user store.

If this setting is not configured here, the value from the .ini file is used.
If this setting is not configured here or in the .ini file, existing local profiles are used.

This setting specifies the types of profile migrated to the user store during logon if a user has no current profile in the user store.

Profile management can migrate existing profiles “on the fly” during logon if a user has no profile in the user store. After this, the user store profile is used by Profile management in both the current session and any other session configured with the path to the same user store.

By default, both local and roaming profiles are migrated to the user store during logon.

To specifies the types of profile migrated to the user store during logon, choose one of the following options:

- Local and roaming profiles
- Local
- Roaming
- None (Disabled)

If you select None, the system uses the existing Windows mechanism to create new profiles, as if in a environment where Profile management is not installed.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, existing local and roaming profiles are migrated.

This setting specifies the path to the profile you want Profile management to use as a template to create new user profiles.

The specified path must be the full path to the folder containing the NTUSER.DAT registry file and any other folders and files required for the template profile.

Note: Ensure that you do not include NTUSER.DAT in the path. For example, with the file \myservername\myprofiles\template\ntuser.dat, set the location as \myservername\myprofiles\template.

Use absolute paths, which can be either UNC paths or paths on the local machine. Use the latter, for example, to specify a template profile permanently on a Citrix Provisioning Services image. Relative paths are not supported.

Note: This setting does not support expansion of Active Directory attributes, system environment variables, or the %USERNAME% and %USERDOMAIN% variables.

By default, this setting is disabled and new user profiles are created from the default user profile on the device where a user first logs on.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no template is used.

This setting enables the template profile to override the local profile when creating new user profiles.

If a user has no Citrix user profile, but a local Windows user profile exists, by default the local profile is used (and migrated to the user store, if this is not disabled). Enabling this policy setting allows the template profile to override the local profile used.
when creating new user profiles.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no template is used.

This setting enables the template profile to override a roaming profile when creating new user profiles.

If a user has no Citrix user profile, but a roaming Windows user profile exists, by default the roaming profile is used (and migrated to the user store, if this is not disabled). Enabling this policy setting allows the template profile to override the roaming profile used when creating new user profiles.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no template is used.

This setting enables Profile management to use the template profile as the default profile for creating all new user profiles.

By default, this setting is disabled and new user profiles are created from the default user profile on the device where a user first logs on.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no template is used.
Registry policy settings

Dec 13, 2013
The Registry section contains policy settings for configuring which Registry keys are included in or excluded from Profile management processing.

This setting specifies the list of registry keys in the HKCU hive excluded from Profile management processing when a user logs off.

When enabled, keys specified in this list are excluded from processing when a user logs off.

By default, this setting is disabled, and all registry keys in the HKCU hive are processed when a user logs off.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no registry keys are excluded from processing.

This setting specifies the list of registry keys in the HKCU hive included in Profile management processing when a user logs off.

When enabled, only keys specified in this list are processed when a user logs off.

By default, this setting is disabled, and all registry keys in the HKCU hive are processed when a user logs off.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, all of HKCU is processed.
Streamed user profiles policy settings

Dec 13, 2013

The Streamed user profiles section contains policy settings for configuring how Profile management processes streamed user profiles.

This setting specifies whether or not Profile management caches streamed files as soon as possible after a user logs on. Caching files after a user logs on saves on network bandwidth, enhancing the user experience.

Use this setting in conjunction with the Profile streaming setting.

By default, this setting is disabled and streamed files are not cached as soon as possible after a user logs on.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, it is disabled.

This setting specifies a lower limit, in megabytes, on the size of files that are streamed. Profile management caches any files this size or larger as soon as possible after a user logs on.

By default, this is set to 0 (zero) and the cache entire profile feature is used. When the cache entire profile feature is enabled, Profile management fetches all profile contents in the user store, after a user logs on, as a background task.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, it is disabled.

This setting enables and disables the Citrix streamed user profiles feature. When enabled, files and folders contained in a profile are fetched from the user store to the local computer only when they are accessed by users after they have logged on. Registry entries and any files in the pending area are fetched immediately.

By default, profile streaming is disabled.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, it is disabled.

This setting specifies which user profiles within an OU are streamed, based on Windows user groups.

When enabled, only user profiles within the specified user groups are streamed. All other user profiles are processed normally.

By default, this setting is disabled and all user profiles within an OU are processed normally.

If this setting is not configured here, the value from the .ini file is used.
If this setting is not configured here or in the .ini file, all user profiles are processed.

This setting specifies a period of time (in days) after which users’ files are written back to the user store from the pending area in the event that the user store remains locked when a server becomes unresponsive. This prevents bloat in the pending area and ensures the user store always contains the most up-to-date files.

By default, this is set to 1 (one) day.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, the default value is used.
Virtual Delivery Agent policy settings

Feb 17, 2014

The Virtual Delivery Agent (VDA) section contains policy settings you can configure to control communication between the VDA and controllers for a site.

Important: The VDA requires information provided by these settings to register with a controller, if you are not using the auto-update feature. Because this information is required for registration, you must configure the following settings using Active Directory's Group Policy Editor, unless you provide this information during the VDA install:

- Controller registration IPv6 netmask
- Controller registration port
- Controller SIDs
- Controllers
- Only use IPv6 controller registration
- Site GUID

This policy setting allows administrators to restrict the VDA to only a preferred subnet (rather than a global IP, if one is registered). This setting specifies the IPv6 address and network where the VDA will register. The VDA will register only on the first address that matches the specified netmask. This setting is valid only if the Only use IPv6 controller registration policy setting is enabled.

By default this setting is blank.

Use this setting only if the Enable auto update of controllers setting is disabled.

This setting specifies the TCP/IP port number the VDA uses to register with a controller, when using registry-based registration.

By default, the port number is set to 80.

Use this setting only if the Enable auto update of controllers setting is disabled.

This setting specifies a space-separated list of controller Security Identifiers (SIDs) the VDA uses to register with a controller, when using registry-based registration. This is an optional setting, that may be used in conjunction with the Controllers setting, to restrict the list of controllers used for registration.

By default, this setting is blank.

Use this setting only if the Enable auto update of controllers setting is disabled.

This setting specifies a space-separated list of controller Fully Qualified Domain Names (FQDNs) the VDA uses to register with a controller, when using registry-based registration. This is an optional setting, that may be used in conjunction with the Controller SIDs setting.
By default, this setting is blank.

This setting enables the VDA to register with a controller automatically after installation.

After the VDA registers, the controller with which it registered sends a list of the current controller FQDNs and SIDs to the VDA. The VDA writes this list to persistent storage. Each controller also checks the site database every 90 minutes for controller information; if a controller has been added or removed since the last check, or if a policy change has occurred, the controller sends updated lists to its registered VDAs. The VDA will accept connections from all the controllers in the most recent list it received.

By default, this setting is enabled.

This setting controls which form of address the VDA uses to register with the controller:

- When enabled, the VDA registers with the controller using the machine's IPv6 address. When the VDA communicates with the controller, it uses the following address order: global IP address, Unique Local Address (ULA), link-local address (if no other IPv6 addresses are available).
- When disabled, the VDA registers and communicates with the controller using the machine's IPv4 address.

By default, this setting is disabled.

Use this setting only if the Enable auto update of controllers setting is disabled.

This setting specifies the Globally Unique Identifier (GUID) of the site the VDA uses to register with a controller, when using Active Directory-based registration.

By default, this setting is blank.
HDX 3D Pro policy settings

Dec 13, 2013

The HDX 3D Pro section contains policy settings for enabling and configuring the image quality configuration tool for your users. The tool enables users to optimize their use of the available bandwidth by adjusting in real time the balance between image quality and responsiveness.

This setting specifies whether or not users can enable and disable lossless compression using the image quality configuration tool. By default, users are not given the option to enable lossless compression.

When a user enables lossless compression, the image quality is automatically set to the maximum value available in the image configuration tool. By default, either GPU or CPU-based compression can be used, according to the capabilities of the user device and the host computer.

This setting specifies the minimum and maximum values that define the range of image quality adjustment available to users in the image quality configuration tool.

Specify image quality values of between 0 and 100, inclusive. The maximum value must be greater than or equal to the minimum value.
Configure COM Port and LPT Port Redirection settings using the registry

Nov 04, 2015

Policy settings for COM Port and LPT Port Redirection have been deprecated from the Studio, and moved to the registry, and are now located under HKLM\Software\Citrix\GroupPolicy\Defaults\Deprecated on either your Master VDA image or your physical VDA machines.

To enable COM port and LPT port redirection, add new registry keys of type REG_DWORD, as follows:

Caution: Editing the registry incorrectly can cause serious problems that may require you to reinstall your operating system. Citrix cannot guarantee that problems resulting from the incorrect use of Registry Editor can be solved. Use Registry Editor at your own risk. Be sure to back up the registry before you edit it.

<table>
<thead>
<tr>
<th>Registry key</th>
<th>Description</th>
<th>Permitted values</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowComPortRedirection</td>
<td>Allow or prohibit COM port redirection</td>
<td>1 (Allow) or 0 (Prohibit)</td>
</tr>
<tr>
<td>LimitComBw</td>
<td>Bandwidth limit for COM port redirection channel</td>
<td>Numeric value</td>
</tr>
<tr>
<td>LimitComBWPercent</td>
<td>Bandwidth limit for COM port redirection channel as a percentage of total session bandwidth</td>
<td>Numeric value between 0 and 100</td>
</tr>
<tr>
<td>AutoConnectClientComPorts</td>
<td>Automatically connect COM ports from the user device</td>
<td>1 (Allow) or 0 (Prohibit)</td>
</tr>
<tr>
<td>AllowLptPortRedirection</td>
<td>Allow or prohibit LPT port redirection</td>
<td>1 (Allow) or 0 (Prohibit)</td>
</tr>
<tr>
<td>LimitLptBw</td>
<td>Bandwidth limit for LPT port redirection channel</td>
<td>Numeric value</td>
</tr>
<tr>
<td>LimitLptBWPercent</td>
<td>Bandwidth limit for LPT port redirection channel as a percentage of total session bandwidth</td>
<td>Numeric value between 0 and 100</td>
</tr>
<tr>
<td>AutoConnectClientLptPorts</td>
<td>Automatically connect LPT ports from the user device</td>
<td>1 (Allow) or 0 (Prohibit)</td>
</tr>
</tbody>
</table>

After configuring these settings, modify your machine catalogs to use the new master image or updated physical machine. Desktops are updated with the new settings the next time users log off.
Use the Monitor Service OData API

Jun 13, 2013

These topics introduce the Monitor Service application programming interface (API), describe what data is available using the API, and how to access it. Examples are provided to illustrate the API's use. Also explained is how to use SSL to secure the Monitor Service OData API endpoints. These topics are aimed at experienced administrators who are familiar with using OData APIs.

In addition to using the Citrix Director console to display historical data, you can query data using the Monitor Service's API. The API uses a broad framework that provides considerable flexibility for querying data. You can use the API to analyze historical trends for future planning, or to perform detailed troubleshooting of connection and machine failures. You can also use the API to extract information for feeding into other tools and processes; for example, using Microsoft Excel's PowerPivot tables to display the data in different ways. Or, you can build a custom user interface on top of the data that the API provides.

The Monitor Service API uses the Open Data (OData) protocol, which is a Web protocol for querying and updating data, built upon Web technologies such as HTTP. For more information about the OData protocol, see: http://www.odata.org.
Data available using the API

Oct 17, 2013
This topic describes the data that can be accessed using the Monitor Service OData API, and provides a schema diagram.

The Monitor Service schema provides the following types of data:

- data relating to connection failures
- machines in a failure state
- session usage
- logon duration
- load balancing data

The following diagram illustrates the Monitor Service schema.
Accessing data using the API

Jun 13, 2013

The following topic explains how to access data using the Monitor Service OData API.

The Monitor Service API is built on top of SQL Server databases using Windows Communication Foundation (WCF) Data Services that are populated during processing and consolidation. Two endpoints are exposed using WCF with wsHttpBinding. The base address is: http://{dc-host}/Citrix/Monitor/OData/v1. You can also use SSL to secure endpoints; see Securing endpoints using SSL for more information.

1. The Methods endpoint exposes service operations which are used by Citrix Director to retrieve data that requires complex grouping and high performance standards, such as queries on the Dashboard and Trends pages. The Methods API URI is: http://{dc-host}/Citrix/Monitor/OData/v1/Methods

2. The Data endpoint exposes read-only access directly to the database entities and can be accessed using the OData query language. This endpoint allows highly flexible access in terms of filtering and column selection, but does not provide the same performance benefits associated with the highly specific service operations. The Data API URI is: http://{dc-host}/Citrix/Monitor/OData/v1/Data

Authentication

To use the Monitor Service OData API, you must be a XenDesktop administrator. To call the API you require read-only privileges; however, the data returned is determined by XenDesktop administrator roles and permissions. For example, Delivery Group Administrators can call the Monitor Service API but the data they can obtain is controlled by Delivery Group access set up using Citrix Studio. For more information about XenDesktop administrator roles and permissions, see Delegated Administration.

Querying the Data

The Monitor Service API is a REST-based API that can be accessed using an OData consumer. OData consumers are applications that consume data exposed using the OData protocol. OData consumers vary in sophistication, from simple web browsers to custom applications that can take advantage of all the features of the OData Protocol. For more information about OData consumers, see: http://www.odata.org/ecosystem#consumers.

Every part of the Monitor Service data model is accessible and can be filtered on the URL. OData provides a query language in the URL format you can use to retrieve entries from a service. For more information, see: http://msdn.microsoft.com/en-us/library/ff478141.aspx.

The query is processed on the server side and can be filtered further using the OData protocol on the client side.

Note: Enums are not supported in the OData protocol; integers are used in their place. To determine the values returned by the Monitor Service OData API, see Determine enum values.

Categories of Data

The Monitor Service collects a variety of data, including user session usage, user logon performance details, session load balancing details, and connection and machine failure information. Data is aggregated differently depending on its category. Understanding the aggregation of data values presented using the OData Method APIs is critical to interpreting the data. For example:

- Connected Sessions and Machine Failures occur over a period of time, therefore they are exposed as maximums over a time period
LogOn Duration is a measure of length of time, therefore is exposed as an average over a time period.

LogOn Count and Connection Failures are counts of occurrences over a period of time, therefore are exposed as sums over a time period.

**Concurrent Data Evaluation**

Sessions must be overlapping to be considered concurrent. However, when the time interval is 1 minute, all sessions in that minute (whether or not they overlap) are considered concurrent i.e. the size of the interval is so small that the performance overhead involved in calculating the precision is not worth the value added. If the sessions occur in the same hour, but not in the same minute, they are not considered to overlap.
About time and date ranges

Jun 13, 2013

This topic explains how to specify and interpret time and date ranges in the Monitor Service API. You need to read this if you are using the Methods endpoint.

When specifying timestamp ranges in the Monitor Service API, note that the range is inclusive. In other words, the timestamp includes everything that happened during that interval up to the start of the next interval. Specifying the same start and end time yields a single data point at the requested granularity.

This model offers a consistent application of time and date range specifications that scale from minute to monthly granularity. The following table shows some examples.

<table>
<thead>
<tr>
<th>Range (start - end)</th>
<th>Interval/Granularity</th>
<th>Points Returned</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:00 - 1:00</td>
<td>1 minute</td>
<td>1</td>
<td>covers 1:00:00 - 1:00:59</td>
</tr>
<tr>
<td>1:00 - 1:01</td>
<td>1 minute</td>
<td>2</td>
<td>1:00, 1:01 and it covers 1:00:00 - 1:01:59</td>
</tr>
<tr>
<td>1:00 - 2:00</td>
<td>1 minute</td>
<td>61</td>
<td>includes up to 2:00:59</td>
</tr>
<tr>
<td>1:07 - 1:12</td>
<td>1 minute</td>
<td>6</td>
<td>1:07 - 1:12:59</td>
</tr>
<tr>
<td>1:00 - 1:00</td>
<td>5 minutes</td>
<td>1</td>
<td>1:00 - 1:04:59</td>
</tr>
<tr>
<td>1:05 - 1:10</td>
<td>5 minutes</td>
<td>2</td>
<td>1:05 - 1:14:59</td>
</tr>
<tr>
<td>1:00 - 2:00</td>
<td>1 hour</td>
<td>2</td>
<td>includes 1:00 - 2:59</td>
</tr>
<tr>
<td>1/1/2012 - 1/2/2012</td>
<td>1 day</td>
<td>2</td>
<td>1/1 and 1/2</td>
</tr>
<tr>
<td>1/1/2012 - 12/12/2012</td>
<td>1 month</td>
<td>12</td>
<td>Data that occurred in each month</td>
</tr>
</tbody>
</table>
Securing endpoints using SSL

Oct 21, 2013

This topic explains how to use SSL to secure the Monitor Service OData API endpoints. If you choose to use SSL, you must configure SSL on all Delivery Controllers in the site; you cannot use a mixture of SSL and non-SSL.

To secure Monitor Service endpoints using SSL, you must perform the following configuration. Some steps need to be done only once per site, others must be run from every machine hosting the Monitor Service in the site. The steps are described below.

**Part 1: Certificate registration with the system**

1. Create a certificate using a trusted certificate manager. The certificate must be associated with the port on the machine that you wish to use for OData SSL.
2. Configure the Monitor Service to use this port for SSL communication. The steps depend on your environment and how this works with certificates. The following example shows how to configure port 449:

   - Associate the certificate with a port:
     ```bash
     netsh http add sslcert ipport=0.0.0.0:449 certhash=97bb629e50d556c80528f4991721ad4f28fb74e9
     appid='{00000000-0000-0000-0000-000000000000}')
     ```
   
   - **Tip:** In a PowerShell command window, ensure you put single quotes around the GUID in the appID, as shown above, or the command will not work. Note that a line break has been added to this example for readability only.

**Part 2: Modify the Monitor Service configuration settings**

1. From any Delivery Controller in the site, run the following PowerShell commands once. This removes the Monitor Service registration with the Configuration Service.
   ```powershell
   asnp citrix.*
   $serviceGroup = get-configregisteredserviceinstance -servicetype Monitor | Select -First 1 ServiceGroupUid
   remove-configserviceGroup -ServiceGroupUid $serviceGroup.ServiceGroupUid
   ```

   - Do the following on all Controllers in the site:
     - Using a cmd prompt, locate the installed Citrix Monitor directory (typically in C:\Program Files\Citrix\Monitor\Service).
       - Within that directory run:
         ```bash
         Citrix.Monitor.Exe -CONFIGUREFIREWALL -ODataPort 449 -RequireODataSsl
         ```
     - Run the following PowerShell commands:
       ```powershell
       asnp citrix.* (if not already run within this window)
       ```

   ```powershell
   get-MonitorServiceInstance | register-ConfigServiceInstance
   Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-MonitorServiceGroupMembership
   ```
Methods

The following topics describe the methods exposed by the API. Each method is described, together with the parameters provided, and what is returned by the API call. Example queries are also shown.

For more information about specifying time intervals, see About time and date ranges. For more information about the values returned by the Monitor Service OData API, see Determine enum values.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetConcurrentSessionsTrend</td>
<td>Gets a trend of concurrent sessions.</td>
</tr>
<tr>
<td>GetConnectionFailureTrendsByType</td>
<td>Gets a list of connection failure counts by type.</td>
</tr>
<tr>
<td>GetConnectionFailureTrendsByTypeLatest</td>
<td>Gets a list of the latest connection failure counts by type.</td>
</tr>
<tr>
<td>GetMachineFailureTrendsByType</td>
<td>Gets a list of machine failure/success counts by type.</td>
</tr>
<tr>
<td>GetLogOnDurationTrend</td>
<td>Gets the logon duration trend.</td>
</tr>
<tr>
<td>GetAverageLogOnDuration</td>
<td>Gets the average logon duration trend over a specified time.</td>
</tr>
<tr>
<td>GetSessionCountTrend</td>
<td>Gets a trend of concurrent connected sessions.</td>
</tr>
<tr>
<td>GetConnectionFailureTrend</td>
<td>Gets the connection failure trend over a specified time.</td>
</tr>
<tr>
<td>GetMachineFailureTrend</td>
<td>Gets the concurrent machine failure trend over a specified time.</td>
</tr>
<tr>
<td>GetAverageLogOnBreakdown</td>
<td>Gets the average logon duration for the specified time, user ID and desktopGroup Uid.</td>
</tr>
<tr>
<td>GetLogOnDurationDetailsTrend</td>
<td>Gets the logon duration trend over a specified time.</td>
</tr>
<tr>
<td>GetLogOnCountTrendFiltered</td>
<td>Gets the logon count trend over a specified time.</td>
</tr>
<tr>
<td>GetLogOnDurationAverageTrend</td>
<td>Gets the logon duration average trend over a specified time, with averages grouped by the specified interval.</td>
</tr>
<tr>
<td>GetEffectiveLoadIndexTrend</td>
<td>Gets the trend of load indices over a specified time. Each data point represents the average load index over the interval. An IQueryable of LoadIndexTrendItems is returned, which represents the effective load index and the breakdown by individual types.</td>
</tr>
<tr>
<td>GetLoadIndexTrend</td>
<td>Gets the load index trend for the specified load index types. An IQueryable of LoadIndexTrends is returned.</td>
</tr>
<tr>
<td>GetConnectedUsersTrend</td>
<td>Gets the connected users trend for Server OS Machines over a specified time.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>GetLastSessionLogOnBreakdown</td>
<td>Gets the breakdown of a specific logon, based on session Uid.</td>
</tr>
</tbody>
</table>
GetAverageLogOnBreakdown

Mar 12, 2015

Gets the average logon duration for the specified time period, user ID and desktopGroupUid.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>DateTime</td>
<td>Start of the time window to get the average for. Must be UTC DateTime.</td>
</tr>
<tr>
<td>endDate</td>
<td>DateTime</td>
<td>End of the time window to get the average for. Must be UTC DateTime.</td>
</tr>
<tr>
<td>userSid</td>
<td>String</td>
<td>User Sid to limit the average to.</td>
</tr>
<tr>
<td>desktopGroupUid</td>
<td>Guid</td>
<td>DesktopGroupUid to limit the average to.</td>
</tr>
</tbody>
</table>

Returns

An IQueryable of LogOnBreakdown objects that contain the following for each logon step:

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>LogonStepItems</td>
<td>List&lt;LogOnStepItem&gt;</td>
<td>List of LogOnStepItems. Each LogonStepItem has LogonStep (Values can be Brokering=1, VMStart=2, HDX=3, Authentication=4, Gpos=5, LogonScripts=6, ProfileLoad=7, Interactive=8, Total=0) and Duration (Nullable&lt;double&gt;) in milliseconds. If no data is available for the step, Duration is null.</td>
</tr>
<tr>
<td>BreakdownType</td>
<td>int</td>
<td>Type of breakdown. UsersLastSession=1, UsersSessionAverage=2, DesktopGroupAverage=3</td>
</tr>
</tbody>
</table>

Note: The HDX duration is a new metric that requires changes to the ICA protocol. This means that, if the new version of the client is not being used, the metrics returned are NULL.

Example

Gets the average logon duration for March 15th, for this user ID and desktopGroupUid.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetAverageLogOnBreakdown?startDate=datetime'2012-03-15T00:00:00'&endDate=2012-03-30T00:00:00'&userSid='User%20Sid'&desktopGroupUid=guid'28D15C1F-7D4C-43BC-BAF2-5886D8514642'&$expand=LogOnStepItems
GetAverageLogOnDuration

Jul 24, 2013

Gets the average logon duration trend for the specified time period. Includes only logons that completed successfully (which have both a LogOnStartDate and a LogOnEndDate in the database).

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>DateTime</td>
<td>Start of the time window to be queried for the average logon duration trend. Must be UTC.</td>
</tr>
<tr>
<td>end</td>
<td>DateTime</td>
<td>End of the time window to be queried for the connection failure trend. Must be UTC.</td>
</tr>
</tbody>
</table>

Returns

Double representing the average logon duration for the specified time period.

Example

Gets the average logon duration over the hour specified for all Delivery Groups.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetAverageLogOnDuration?startDate=datetime'2012-06-05T12:24:00'&endDate=datetime'2012-06-05T13:24:00'
GetConcurrentSessionsTrend

July 24, 2013

Gets data points to represent a trend of concurrent sessions as a list of DateTime/value pairs for the specified time period. Each point represents the maximum number of concurrent sessions over the interval specified. The last point in the trend can represent the number of current users in the system.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>DateTime</td>
<td>Start of the time window to be queried for concurrent sessions. Must be UTC. Inclusive.</td>
</tr>
<tr>
<td>endDate</td>
<td>DateTime</td>
<td>End of the time window to be queried for concurrent sessions. Must be UTC. Inclusive.</td>
</tr>
<tr>
<td>intervalLength</td>
<td>int</td>
<td>Interval between data points in minutes for the trend.</td>
</tr>
</tbody>
</table>

Returns

IQueryable list of TrendItem objects - IQueryable<TrendItem> ordered by Date ascending.

Each TrendItem contains a UTC Date (DateTime) property and a Value property (double) for plotting the trend.

Example

Gets a one hour trend (data point every minute) of the concurrent sessions:

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetConcurrentSessionsTrend?startDate=datetime'2012-06-05T12:26:00'&endDate=datetime'2012-06-05T13:26:00'&intervalLength=1
GetConnectedUsersTrend

Jul 24, 2013

Gets the connected users trend for Server OS Machines for the specified time period. Each data point in the trend has the total number of users connected to Server OS Machines for the last interval (intervalLength).

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>DateTime</td>
<td>Start of the time window to be queried for the connected users trend. Must be UTC.</td>
</tr>
<tr>
<td>endDate</td>
<td>DateTime</td>
<td>End of the time window to be queried for the connected users trend. Must be UTC.</td>
</tr>
<tr>
<td>intervalLength</td>
<td>int</td>
<td>Interval between data points in minutes for the trend.</td>
</tr>
<tr>
<td>desktopGroupsFilter</td>
<td>string</td>
<td>Comma-separated list of Delivery Group Uids to limit the trend result to.</td>
</tr>
<tr>
<td>machineFilter</td>
<td>string</td>
<td>Comma-separated list of machine Sids to limit the trend result to.</td>
</tr>
</tbody>
</table>

Returns

An IQueryable of objects that contains the following:

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConnectedUsersTrend</td>
<td>List&lt;TrendItem&gt;</td>
<td>TrendItem contains a UTC Date (DateTime) and a Count (int) for the maximum number of connected users in the interval.</td>
</tr>
</tbody>
</table>

Examples

Gets a 24-hour trend (data point every 15 minutes) of the connected users trend for the two Delivery Groups specified.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetConnectedUsersTrend?startDate=datetime'2011-09-30T00:00:00'&endDate=dateTime'2011-10-01T00:00:00'&intervalLength=15&desktopGroupsFilter='04D53BBE-F5CC-11E0-B401-3F054824019B,0FBEFBBE-F5CC-11E0-9472-40054824019B'

Gets a week-long trend (data point every 4 hours) of the connected users trend.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetConnectedUsersTrend?startDate=datetime'2011-09-23T00:00:00'&endDate=dateTime'2011-10-01T00:00:00'&intervalLength=240
GetConnectionFailureTrend

Gets the connection failure trend for the specified time period. Each data point in the trend represents the total number of connection failures in the last interval [intervalLength parameter].

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>DateTime</td>
<td>Start of the time window to be queried for the connection failure trend. Must be UTC.</td>
</tr>
<tr>
<td>endDate</td>
<td>DateTime</td>
<td>End of the time window to be queried for the connection failure trend. Must be UTC.</td>
</tr>
<tr>
<td>intervalLength</td>
<td>int</td>
<td>Interval between data points in minutes for the trend.</td>
</tr>
<tr>
<td>desktopGroupsFilter</td>
<td>string</td>
<td>Comma-separated list of Delivery Group uids to limit the trend result to.</td>
</tr>
<tr>
<td>connectionFailureTypeFilter</td>
<td>string</td>
<td>Comma-separated list of connection failure type ints to limit the trend result to. This integer corresponds to ConnectionFailureType enum.</td>
</tr>
<tr>
<td>sessionSupportFilter</td>
<td>string</td>
<td>Comma-separated list of machine type ints (Single=1, Multiple=2, Unknown=0) to limit the trend result to. No value returns both types.</td>
</tr>
</tbody>
</table>

Returns

An IQueryable of TrendItem objects

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>FailureTrend</td>
<td>List&lt;TrendItem&gt;</td>
<td>TrendItem contains a UTC Date (DateTime) and a Count(int) of failures to indicate the trend of connection failures over the specified time period for the filters specified.</td>
</tr>
</tbody>
</table>

Examples

Gets a 24-hour trend (data point every 15 minutes) for all types of connection failures for the two Delivery Groups specified with Desktop OS Machine types.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetConnectionFailureTrend?start=datetime'2011-09-30T00:00:00'&end=datetime'2011-10-01T00:00:00'intervalLength=240&connectionFailureTypeFilter=

Gets a week-long trend (data point every 4 hours) for 3 types of connection failures (aggregated) for all Delivery Groups the user has access to.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetConnectionFailureTrend?start=datetime'2011-09-23T00:00:00'&end=datetime'2011-10-01T00:00:00'intervalLength=240&connectionFailureTypeFilter='2,3,4'
GetConnectionFailureTrendsByType
Jul 24 , 20 13

Gets a list of connection failure counts by type, together with a trend for each type, for the speciﬁed time period. Each data point in the trend represents the total number of connection failures in
the last interval (intervalLength parameter). Provides ability to specify a particular failure type to limit the query or get all failure types.
P aramet er Name

T ype

Comment s

startDate

DateT ime

Start of the time window to be queried for the connection failure trend. Must be UT C.

endDate

DateT ime

End of the time window to be queried for the connection failure trend. Must be UT C.

intervalLength

Int

Interval between data points in minutes for the trend.

connectionFailureT ype

Int

[ConnectionFailureT ype] If provided (>0), returns only that connection failure type in the results. Otherwise (null or zero) returns a list of all failure types. T his
integer corresponds to ConnectionFailureT ype enum.

Ret urns
An IQueryable of ConnectionFailureTrend objects that contain the following for each ConnectionFailureType.
If all types are requested, then the sum is also returned with ConnectionFailureType = -1
If $expand=FailureTrend is included in the query, then the trend and TotalFailureCount properties are retrieved for each failure type.
If $expand=DesktopGroupBreakdown is included in the query, that property is also retrieved for each failure type.
P ropert y Name

T ype

Comment s

ConnectionFailureT ype

int [ConnectionFailureT ype]

T ype of connection failure (from enum ConnectionFailureT ype).

FailureT rend

List<T rendItem>

T rendItem contains a UT C Date (DateT ime), and a Value (nullable double) to indicate the trend of connection failures over the
time period specified for this type of failure.

T otalFailureCount

int

T otal number of failures over the specified time period.

DesktopGroupBreakdown

List<DesktopGroupBreakdown>

DesktopGroupBreakdown contains a DesktopGroup object (must be included in the expand to be returned) and Count (int) to
represent the total number of failures in the specified time window.

Examples
Retrieve both Delivery Group breakdown and the failure trend for all failure types for Nov-14 at 5am (UTC) to Nov 14 at 7am (UTC) with a data point every 30 minutes:

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetConnectionFailureTrendsByType?startDate=datetime'2011-11-14T05:00:00'&endDate=datetime'2011-11-14T07:00:00'&intervalLength=30&connectionFailureTyp
Retrieve Delivery Group breakdown only for the failure type 4 (No Capacity Available) for Nov-14 at 5am (UTC) to Nov 14 at 7am (UTC)

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetConnectionFailureTrendsByType?startDate=datetime'2011-11-14T05:00:00'&endDate=datetime'2011-11-14T07:00:00'&connectionFailureType=4&$expand=Des
Retrieve the failure trend for all failure types from Nov-14 at 12am (UTC) to Nov 14 at 11pm (UTC) with a data point every hour:

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetConnectionFailureTrendsByType?startDate=datetime'2011-11-14T00:00:00'&endDate=datetime'2011-11-14T23:00:00'&intervalLength=60&connectionFailureTyp

https://docs.citrix.com

© 1999-2017 Citrix Systems, Inc. All rights reserved.

p.2678


GetConnectionFailureTrendsByTypeLatest

Nov 11, 2013

Gets a list of connection failure counts by type, together with a trend for each type, for the specified number of intervals from now (UTC server time). Each data point in the trend represents the total number of connection failures in the last interval (intervalLength parameter). Provides ability to specify a particular failure type to limit the query or get all failure types.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>intervalLength</td>
<td>int</td>
<td>Interval between data points in minutes for the trend.</td>
</tr>
<tr>
<td>numberOfIntervals</td>
<td>int</td>
<td>Number of requested data points. Used with intervalLength to calculate StartDate.</td>
</tr>
<tr>
<td>connectionFailureType</td>
<td>int [ConnectionFailureType]</td>
<td>If provided (&gt;0), returns only that connection failure type in the results. Otherwise (null or zero) returns a list of all failure types. This integer corresponds to ConnectionFailureType enum.</td>
</tr>
</tbody>
</table>

Returns:

An IQueryable of ConnectionFailureTrend objects that contain the following for each ConnectionFailureType.

If all types are requested, then the sum is also returned with ConnectionFailureType = -1

If $expand=FailureTrend is included in the query, the trend and TotalFailureCount properties are retrieved for each failure type.

If $expand=DesktopGroupBreakdown is included in the query, that property is also retrieved for each failure type.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConnectionFailureType</td>
<td>int [ConnectionFailureType]</td>
<td>Type of connection failure (from enum ConnectionFailureType).</td>
</tr>
<tr>
<td>FailureTrend</td>
<td>List&lt;TrendItem&gt;</td>
<td>TrendItem contains a UTC Date (DateTime), and a Value (nullable double) to indicate the trend of connection failures over the time period specified for this type of failure.</td>
</tr>
<tr>
<td>TotalFailureCount</td>
<td>int</td>
<td>Total number of failures over the specified time period.</td>
</tr>
<tr>
<td>DesktopGroupBreakdown</td>
<td>List&lt;DesktopGroupBreakdown&gt;</td>
<td>DesktopGroupBreakdown contains a DesktopGroup object (must be included in the expand to be returned) and Count (int) to represent the total number of failures in the specified time window.</td>
</tr>
</tbody>
</table>

Examples

Retrieve both Delivery Group breakdown and the failure trend for the last 60 minutes with a data point every 1 minute:

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetConnectionFailureTrendsByTypeLatest?$expand=DesktopGroupBreakdown,Trend

Retrieve Delivery Group breakdown only for the failure type 4 (No Capacity Available) for the last 30 minutes:

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetConnectionFailureTrendsByTypeLatest?$expand=DesktopGroupBreakdown,DesktopGroupBreakdown<DesktopGroup>$expand=DesktopGroupBreakdown<DesktopGroup>Count

Retrieve the failure trend for all failure types for the last 24 hours with a data point every hour:

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetConnectionFailureTrendsByTypeLatest?$expand=FailureTrend

https://docs.citrix.com © 1999-2017 Citrix Systems, Inc. All rights reserved. p.2679
GetLastSessionLogOnBreakdown

Gets the logon breakdown for User and Delivery Group. Returns the current User session, if one exists.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Required?</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>userSid</td>
<td>String</td>
<td>Yes</td>
<td>The ID of the user for whom information is being requested.</td>
</tr>
<tr>
<td>desktopGroupUid</td>
<td>Guid</td>
<td>Yes</td>
<td>Delivery Group being queried.</td>
</tr>
<tr>
<td>machineUid</td>
<td>String</td>
<td>No</td>
<td>The machine on which the last session was run. This may not be available.</td>
</tr>
</tbody>
</table>

Returns

An IQueryable of LogOnBreakdown objects that contain the following for each logon step:

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>LogonStepItems</td>
<td>List&lt;LogOnStepItem&gt;</td>
<td>List of LogOnStepItems. Each LogonStepItem has LogonStep (Values can be Brokering=1, VMStart=2, HDX=3, Authentication=4, Gpos=5, LogonScripts=6, ProfileLoad=7, Interactive=8, Total=0) and Duration (Nullable&lt;double&gt;) in milliseconds. If no data is available for the step, Duration is null.</td>
</tr>
<tr>
<td>BreakdownType</td>
<td>int</td>
<td>Type of breakdown - UsersLastSession=1, UsersSessionAverage=2, DesktopGroupAverage=3</td>
</tr>
</tbody>
</table>

Example

Get logon breakdown for User and Delivery Group. Returns the current User session if one exists.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetLastSessionLogOnBreakdown?userSid='User%20Sid'&desktopGroupUid=guid'28D15C1F-7D4C-43BC-BAF2-5886D8514642'&machineUid=guid'28D15C1F-7D4C-43BC-BAF2-5886D8514642'
GetEffectiveLoadIndexTrend

Gets the Load Evaluation Index (LEI) trend for Server OS Machines for the specified time period (in minutes). Each data point represents the average load index over this interval. An IQueryable of LoadIndexTrendItems is returned, which represents the effective load index and the breakdown by individual types (CPU, Memory, Disk, Network, SessionCount).

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>DateTime</td>
<td>Start of the time window to be queried for the load index trend. Must be UTC.</td>
</tr>
<tr>
<td>endDate</td>
<td>DateTime</td>
<td>End of the time window to be queried for the load index trend. Must be UTC.</td>
</tr>
<tr>
<td>intervalLength</td>
<td>int</td>
<td>Interval between data points in minutes for the trend.</td>
</tr>
<tr>
<td>desktopGroupsFilter</td>
<td>string</td>
<td>Comma-separated list of Delivery Group Uids to limit the trend result to.</td>
</tr>
<tr>
<td>machineFilter</td>
<td>string</td>
<td>Comma-separated list of machine Sids to limit the trend result to. Expects MultiSession machines.</td>
</tr>
</tbody>
</table>

Returns

An IQueryable of objects that contains the following:

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoadIndexTrend</td>
<td>List&lt;LoadIndexTrendItem&gt;</td>
<td>LoadIndexTrendItem contains a UTC Date (DateTime) and an integer for total LEI, as well as integers for each of the LEI portions (CPU, Memory, Disk, Network, SessionCount) in a separate object (provided if $expand=Breakdown is included in the query).</td>
</tr>
</tbody>
</table>

Examples

Gets a 24-hour trend (data point every 15 minutes) of the load index trend and breakdown for the two Delivery Groups specified.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetEffectiveLoadIndexTrend?startDate=datetime'2011-09-30T00:00:00'&endDate=datetime'2011-10-01T00:00:00'&intervalLength=15&desktopGroupsFilter

Gets a week-long trend (data point every 4 hours) of the load index trend.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetEffectiveLoadIndexTrend?startDate=datetime'2011-09-23T00:00:00'&endDate=datetime'2011-10-01T00:00:00'&intervalLength=240&$expand=Breakdown
GetLoadIndexTrend

Yulu 24, 2013

Gets the Load Evaluation Index (LEI) trend for Server OS Machines for the specified time period, limited by load index types. Each data point in the trend has the total LEI, as well as the breakdown for each load index type (CPU, Memory, Disk, Network, SessionCount).

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>DateTime</td>
<td>Start of the time window to be queried for the load index trend. Must be UTC.</td>
</tr>
<tr>
<td>endDate</td>
<td>DateTime</td>
<td>End of the time window to be queried for the load index trend. Must be UTC.</td>
</tr>
<tr>
<td>intervalLength</td>
<td>int</td>
<td>Interval between data points in minutes for the trend.</td>
</tr>
<tr>
<td>desktopGroupsFilter</td>
<td>string</td>
<td>Comma-separated list of Delivery Group Uids to limit the trend result to.</td>
</tr>
<tr>
<td>machineFilter</td>
<td>string</td>
<td>Comma-separated list of machine Sids to limit the trend result to. Expects MultiSession machines.</td>
</tr>
<tr>
<td>indexTypeFilter</td>
<td>string</td>
<td>Comma-separated list of load index to include (effective, CPU, network, memory, disk, session count). Corresponds to the LoadIndexType enum. An empty or null string returns all.</td>
</tr>
</tbody>
</table>

**Returns**

An IQueryable of objects that contains the following for each index type specified:

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoadIndexType</td>
<td>int</td>
<td>[LoadIndexType] Corresponds to the LoadIndex type enum.</td>
</tr>
<tr>
<td>LoadIndexTrend</td>
<td>List&lt;TrendItem&gt;</td>
<td>TrendItem contains a UTC Date (DateTime) and an integer for the index.</td>
</tr>
</tbody>
</table>

**Examples**

Gets a 24-hour trend (data point every 15 minutes) of the load index trend and breakdown for the two Delivery Groups specified.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetLoadIndexTrend?startDate=datetime'2011-09-30T00:00:00'&endDate=datetime'2011-10-01T00:00:00'intervalLength=15&desktopGroupsFilter='04D53BBE-F5CC-11E0-B401-3F054824019B,0FBEFBBE-F5CC-11E0-9472-40054824019B'

Gets a week-long trend (data point every 4 hours) of the load index trend and breakdown.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetLoadIndexTrend?startDate=datetime'2011-09-23T00:00:00'&endDate=datetime'2011-10-01T00:00:00'intervalLength=240
GetLogOnCountTrend

Sep 02, 2015

Gets the logon duration trend (average) grouped by an interval (TimeOfDay, DayOfWeek, etc) over the specified time period. Includes only logons that completed successfully (which have both a LogOnStartDate and a LogOnEndDate in the database).

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>DateTime</td>
<td>Start of the time window to be queried for the average logon duration trend. Must be UTC.</td>
</tr>
<tr>
<td>end</td>
<td>DateTime</td>
<td>End of the time window to be queried for the logon duration trend. Must be UTC.</td>
</tr>
<tr>
<td>intervalLength</td>
<td>int</td>
<td>Interval between data points in minutes for the trend.</td>
</tr>
</tbody>
</table>

Returns

An IQueryable of TrendItem objects. TrendItem contains a UTC Date (DateTime) and a Value to indicate the trend of average logon counts over the start and end time period specified.

Example

Gets one hour of data, single data point per minute representing the total number of logons for that minute. If no logons occurred during that minute, the value in the TrendItem is zero.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetLogOnCountTrend?startDate=DateTime'2012-06-05T12:24:00'&endDate=DateTime'2012-06-05T13:24:00'&intervalLength=1
GetLogOnCountTrendFiltered

Sep 02, 2015

Gets the logon count trend for the specified time period. Each data point in the trend represents the total number of logons in the last interval (IntervalLength). Includes only logons that completed successfully (which have both a LogOnStartDate and a LogOnEndDate in the database).

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>DateTime</td>
<td>Start of the time window to be queried for the logon count trend. Must be UTC.</td>
</tr>
<tr>
<td>endDate</td>
<td>DateTime</td>
<td>End of the time window to be queried for the logon count trend. Must be UTC.</td>
</tr>
<tr>
<td>intervalLength</td>
<td>int</td>
<td>Interval between data points in minutes for the trend.</td>
</tr>
<tr>
<td>desktopGroupsFilter</td>
<td>string</td>
<td>Comma-separated list of desktop group Uids to limit the trend result to.</td>
</tr>
</tbody>
</table>

Returns

A list of TrendItems:

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>LogonCountTrend</td>
<td>List&lt;TrendItem&gt;</td>
<td>LogonDurationTrendItem contains a UTC Date (DateTime) and a Count (int) to represent the number of logons for the date.</td>
</tr>
</tbody>
</table>

Example

Gets a 24-hour trend (data point every 15 minutes) of the logon count trend (in milliseconds) for the two Delivery Groups specified.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetLogOnCountTrendFiltered?start=datetime'2011-09-30T00:00:00'&end=dateti ... 00:00:00'intervalLength=15&desktopGroupsFilter='04D53BBE-F5CC-11E0-B401-3F054824019B,0FBEFBBE-F5CC-11E0-9472-40054824019B'

Gets a week-long trend (data point every 4 hours) of the logon duration trend (in milliseconds).

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetLogOnCountTrendFiltered?start=datetime'2011-09-23T00:00:00'&end=datetime'2011-10-01T00:00:00'intervalLength=240
GetLogOnDurationAverageTrend

Get the logon duration average trend for the specified time window, with averages grouped by the specified interval. Each data point in the trend has average logon duration and breakdown for each interval in the groupByInterval.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>DateTime</td>
<td>Start of the time window to be queried for the logon duration trend. Must be UTC.</td>
</tr>
<tr>
<td>end</td>
<td>DateTime</td>
<td>End of the time window to be queried for the logon duration trend. Must be UTC.</td>
</tr>
<tr>
<td>groupByInterval</td>
<td>int</td>
<td>How the averages are rolled up. Options: TimeOfDay=1, DayOfWeek=2, Month=3, Year=4. Required.</td>
</tr>
<tr>
<td>desktopGroupsFilter</td>
<td>string</td>
<td>Comma-separated list of Delivery Group Uids to limit the trend result to.</td>
</tr>
</tbody>
</table>

Returns

A list of GroupedLogOnDurationTrendItem

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
</table>
| GroupByDate         | int                    | Represents the group by date (4=4am, 2007=year 2007 depending on the groupByInterval that was input). Number of points returned: 
GroupByInterval=TimeOfDay (1) 24 points (0-23)
0=12AM, 1=AM, 2=2AM ... 23=11PM
GroupByInterval=DayOfWeek (2) 7 points (0-6)
0=Sunday, 1=Monday ... 6=Saturday
GroupByInterval=Month (3) 12 points (0-11)
0=January, 1=February ... 11=December
GroupByInterval=Year (4) However many years are included in the startDate/endDate range. Order of Points: The data is always returned ordered by the GroupByDate, regardless of the start and end dates passed to the API. |
| Value               | Nullable<double>      | Average total logon duration in milliseconds and the average duration of each step for this group. |
| Breakdown           | Collection<LogOnStepItem> | Collection of LogOnStepItem objects which each contain: Duration (Nullable<double>) which has the average duration of this step in milliseconds and LogOnStep (int) which returns the integer enum value of the logon step (refers to the LogOnStep enum) |

Example

Gets a 7-day average by time of day (hour) of the logon duration trend (in milliseconds) for the two Delivery Groups specified. This returns a trend item for each hour of the day that has the average logon duration for that time of day over the specified time frame (24 data points returned).

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetDurationAverageTrend?start=datet ime'2011-09-23T00:00:00'&end=datet ime'2011-09-30T00:00:00'&groupByInterval=1&desktopGroupsFilter='04D53BBE-F5CC-11E0-B401-3F054824019B,0FBEFBBE-F5CC-11E0-9472-40054824019B
GetLogOnDurationDetailsTrend

Gets the logon duration trend for the specified time period. Each data point in the trend has the total logon duration, as well as the breakdown for each step.

### Parameters

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>DateTime</td>
<td>Start of the time window to be queried for the connection failure trend. Must be UTC.</td>
</tr>
<tr>
<td>endDate</td>
<td>DateTime</td>
<td>End of the time window to be queried for the connection failure trend. Must be UTC.</td>
</tr>
<tr>
<td>intervalLength</td>
<td>int</td>
<td>Interval between data points in minutes for the trend.</td>
</tr>
<tr>
<td>desktopGroupsFilter</td>
<td>string</td>
<td>Comma-separated list of Delivery Group Uids to limit the trend result to.</td>
</tr>
</tbody>
</table>

### Returns

A list of LogonDurationTrendItems:

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>LogonDurationTrend</td>
<td>List&lt;LogonDurationTrendItem&gt;</td>
<td>LogOnDurationTrendItem contains a UTC Date (DateTime) and a double for average logon duration in milliseconds for that time. Also contains a breakdown for the average duration of each logon step for each data point.</td>
</tr>
</tbody>
</table>

### Example

Gets a 24-hour trend (data point every 15 minutes) of the logon duration trend (in milliseconds) for the two Delivery Groups specified.

```
http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetLogOnDurationDetailsTrend?startDate=2011-09-30T00:00:00'&endDate=2011-10-01T00:00:00'intervalLength=15&desktopGroupsFilter='04D53BBE-F5CC-11E0-B401-3F054824019B,0FBEFBBE-F5CC-11E0-9472-40054824019B'
```

Gets a week-long trend (data point every 4 hours) of the logon duration trend (in milliseconds).

```
http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetLogOnDurationDetailsTrend?startDate=2011-09-23T00:00:00'&endDate=2011-10-01T00:00:00'intervalLength=240
```
GetLogOnDurationTrend

Sep 03, 2015

Gets the logon trend (average) over the specified time period.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>DateTime</td>
<td>Start of the time window to be queried for the average logon duration trend. Must be UTC.</td>
</tr>
<tr>
<td>end</td>
<td>DateTime</td>
<td>End of the time window to be queried for the average logon duration trend. Must be UTC.</td>
</tr>
<tr>
<td>intervalLength</td>
<td>int</td>
<td>Interval between data points in minutes for the trend.</td>
</tr>
</tbody>
</table>

Returns

An IQueryable of TrendItem objects. TrendItem contains a UTC Date (DateTime) and a Value (in milliseconds) to indicate average logon duration trend over the specified time period.

Examples

Gets one hour of data, single data point per minute representing the average logon duration for that minute. If no logons occurred during that minute, the value in the TrendItem is NaN (not a number).

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetLogOnDurationTrend?startDate=datetime'2012-06-05T12:24:00'&endDate=datetime'2012-06-05T13:24:00'&intervalLength=1

Gets single data point representing the average logon duration for that hour.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetLogOnDurationTrend?startDate=datetime'2012-06-05T12:24:00'&endDate=datetime'2012-06-05T13:23:00'&intervalLength=60
GetMachineFailureTrend

Gets the concurrent machine failure trend for the specified time period. Each data point in the trend represents the maximum number of failed machines at any point in the last interval (intervalLength parameter). Provides ability to specify a particular machine failure type to limit the query or get all failure types.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>DateTime</td>
<td>Start of the time window to be queried for the connection failure trend. Must be UTC.</td>
</tr>
<tr>
<td>endDate</td>
<td>DateTime</td>
<td>End of the time window to be queried for the connection failure trend. Must be UTC.</td>
</tr>
<tr>
<td>intervalLength</td>
<td>int</td>
<td>Interval between data points in minutes for the trend.</td>
</tr>
<tr>
<td>desktopGroupsFilter</td>
<td>string</td>
<td>Comma-separated list of Delivery Group Uids to limit the trend result to.</td>
</tr>
<tr>
<td>machineFailureTypeFilter</td>
<td>string</td>
<td>Comma-separated list of machine failure type integers to limit the trend result to. This integer corresponds to MachineFaultStateCode enum. If not provided, returns all types aggregated in the trend.</td>
</tr>
<tr>
<td>sessionSupportFilter</td>
<td>int</td>
<td>Comma-separated list of machine types integers (Single=1, Multiple=2, Unknown=0) to limit the trend result to. No value returns all types</td>
</tr>
</tbody>
</table>

Returns

A list of TrendItem objects (FailureTrend).

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>FailureTrend</td>
<td>List&lt;TrendItem&gt;</td>
<td>TrendItem contains a UTC Date (DateTime) and a Count (int) of failures to indicate the trend of machine failures over the specified time period for the filters specified.</td>
</tr>
</tbody>
</table>

Example

Gets a 24-hour trend (data point every 15 minutes) for all types of machine failures for the two Delivery Groups specified with Desktop OS Machine types.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetMachineFailureTrend?start=datetime'2011-09-30T00:00:00'&end=datetime'2011-10-01T00:00:00'&intervalLength=15&desktopGroupsFilter='04D53BBE-F5CC-11E0-B401-3F054824019B,0FBEFBBE-F5CC-11E0-9472-40054824019B'

Gets a 24-hour trend (data point every 2 hours) for all Delivery Groups, all Session Types, and for machine failures of type "Failed to Start" and "Stuck on Boot".

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetMachineFailureTrend?start=datetime'2011-09-23T00:00:00'&end=datetime'2011-10-01T00:00:00'&intervalLength=240&machineFailureTypeFilter='2,3'
GetMachineFailureTrendsByType

Jul 24, 2013

Gets a list of machine failure and success counts by type, together with a trend for each type, for the specified time period. Each data point in the trend represents the total number of machines in a failure state in the last interval (intervalLength parameter). You can specify a particular failure type to limit the query or get all failure types.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>DateTime</td>
<td>Start of the time window to be queried for the machine failure trend. Must be UTC. Not required if not fetching trends.</td>
</tr>
<tr>
<td>endDate</td>
<td>DateTime</td>
<td>End of the time window to be queried for the machine failure trend. Must be UTC.</td>
</tr>
<tr>
<td>intervalLength</td>
<td>int</td>
<td>Interval between data points in minutes for the trend.</td>
</tr>
<tr>
<td>machineFailureType</td>
<td>int (MachineFailureType)</td>
<td>If provided (&gt;0), returns only that machine failure type in the results. Otherwise (null or zero) returns a list of all failure types. This integer corresponds to MachineFailureType enum.</td>
</tr>
<tr>
<td>sessionSupport</td>
<td>int (SessionSupport)</td>
<td>Nullable int. If not provided, then returns all sessionSupport types. Single=1, Multiple=2, Unknown=0</td>
</tr>
</tbody>
</table>

Returns

An IQueryable of MachineFailureTrend objects that contain the following for each MachineFailureType.

If all types are requested, the sum is also returned with MachineFailureType = -1

If $expand=FailureTrend is included in the query, the trend and TotalFailureCount properties are retrieved for each failure type.

If $expand=DesktopGroupBreakdown is included in the query, that property is also retrieved for each failure type.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>MachineFailureType</td>
<td>int [MachineFailureType]</td>
<td>Type of connection failure (see: Determine enum values).</td>
</tr>
<tr>
<td>TotalFailureCount</td>
<td>int</td>
<td>Total number of failures at the endDate specified.</td>
</tr>
<tr>
<td>FailureTrend</td>
<td>List&lt;TrendItem&gt;</td>
<td>TrendItem contains a UTC Date (DateTime), a Count (double) to indicate the trend of machine failures over the time period specified for the type of failure.</td>
</tr>
<tr>
<td>DesktopGroupBreakdown</td>
<td>List&lt;DesktopGroupBreakdown&gt;</td>
<td>DesktopGroupBreakdown contains a DesktopGroup object and Count (int) to represent the total number of machines in a failure state at the endDate specified.</td>
</tr>
</tbody>
</table>

Examples

Retrieve both Delivery Group breakdown and the failure trend for all machine failure types for Nov-14 at 5am (UTC) to Nov 14 at 7am (UTC), with a data point every 30 minutes for all Sener OS Machines (i.e. sessionSupport = 2):

http://dc-host/Citrix/Monitor/OData/v1/Methods/GetMachineFailureTrendsByType?startDate=datet ime'2011-11-14T05:00:00'&endDate=datetime'2011-11-14T07:00:00'&intervalLength=30&machineFailureType=0&sessionSupport=2

Retrieve Delivery Group breakdown only for machine failure type 3 (Stuck on Boot) for Nov-14 at 5am (UTC) to Nov 14 at 7am (UTC) for Desktop OS Machines:

http://dc-host/Citrix/Monitor/OData/v1/Methods/GetMachineFailureTrendsByType?startDate=datet ime'2011-11-14T05:00:00'&endDate=datetime'2011-11-14T07:00:00'&machineFailureType=3&sessionSupport=1

Retrieve the failure trend for all failure types for all machine session types from Nov-14 at 12am (UTC) to Nov 14 at 11pm (UTC) with a data point every hour:

http://dc-host/Citrix/Monitor/OData/v1/Methods/GetMachineFailureTrendsByType?startDate=datet ime'2011-11-14T00:00:00'&endDate=datet ime'2011-11-14T23:00:00'&intervalLength=60&machineFailureType=0
GetSessionCountTrend

Jul 24, 2013

Gets the data points to represent a trend of concurrent sessions (connected) as a list of DateTime/value pairs for the specified time window. Each point represents the maximum number of concurrent connected sessions over the interval specified. This value is calculated as the sum of the DesktopGroups and the MAX of the sum.

Parameter Name | Type | Comments
---|---|---
startDate | DateTime | Start of the time window to be queried for concurrent sessions. Must be UTC.
dateTime | DateTime | End of the time window to be queried for concurrent sessions. Must be UTC.
intervalLength | int | Interval between data points in minutes for the trend.
sessionSupportFilter | string | List of comma-delimited machine type ints to limit the query to (Single=1, Multiple=2, Unknown=0). If not specified, default is all.
connectionStateFilter | string | List of comma-delimited connection state integers to limit the query to (1=Connected, 2=Disconnected). If not specified, both are returned.
desktopGroupFilter | string | List of comma-delimited Delivery Group Uids to limit the query to. If not specified, returns all Delivery Groups.

Returns

IQueryable list of TrendItem objects - IQueryable<TrendItem> - ordered by Date ascending.

Each TrendItem contains a UTC Date (DateTime) property and a Value property (int) for plotting the trend.

Example

Gets a 24-hour trend with data points every hour for concurrent connected sessions. Each data point represents the maximum number of concurrent sessions (connected only) in the last interval (60 minutes in this case).

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetSessionCountTrend?startDate=datetime'2011-09-30T00:00:00'&endDate=datetime'2011-10-01T00:00:00'&intervalLength=60&connectionStateFilter='1'

Gets a 24-hour trend with data points every hour for concurrent disconnected sessions. Each data point represents the maximum number of concurrent sessions (disconnected only) in the last interval (60 minutes in this case).

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetSessionCountTrend?startDate=datetime'2011-09-30T00:00:00'&endDate=datetime'2011-10-01T00:00:00'&intervalLength=60&connectionStateFilter='2'

Gets past week's trend with data points every 6 hours for concurrent sessions (connected and disconnected) for both machine types (Desktop OS and Server OS Machines). Each data point represents the maximum number of concurrent sessions in the last interval (6 hours in this case).

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetSessionCountTrend?startDate=datetime'2011-09-23T00:00:00'&endDate=datetime'2011-10-01T00:00:00'&intervalLength=360
GetSessionSummary

Jul 24, 2013

Gets the data points to represent a trend of concurrent sessions (connected) as a list of DateTime/value pairs for the specified time period. Each point represents the maximum number of concurrent connected sessions over the interval specified.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>DateTime</td>
<td>Start of the time window to be queried for concurrent sessions. Must be UTC.</td>
</tr>
<tr>
<td>endDate</td>
<td>DateTime</td>
<td>End of the time window to be queried for concurrent sessions. Must be UTC.</td>
</tr>
<tr>
<td>sessionSupportFilter</td>
<td>string</td>
<td>List of comma-delimited machine type integers to limit the query to (Single=1, Multiple=2, Unknown=0). Default is all types, if no filter specified.</td>
</tr>
<tr>
<td>desktopGroupFilter</td>
<td>string</td>
<td>List of comma-delimited Delivery Group Uids to limit the query to. If no value is specified, all Delivery Groups are returned.</td>
</tr>
<tr>
<td>connectionStateFilter</td>
<td>string</td>
<td>List of comma-delimited connection state integers to limit the query to (1=Connected, 2=Disconnected). If not specified, default is both.</td>
</tr>
</tbody>
</table>

Returns

IQueryable list of SessionSummary objects - IQueryable<SessionSummary>.

Each SessionSummary contains the following:

- DesktopGroup (name and Uid)
- AverageSessionCount (double)
- Average number of concurrent sessions for the specified time period. This is the average of the peak values for the time period using the granularity available for that time period
- AverageUserCount (double)
- Average number of users connected for the specified time period
- PeakUserCount (int)
- Peak number of users connected at any point in the specified time period
- UniqueUserCount (int)
- Total number of unique users over the specified time period
- AverageHoursPerDay (double)
- Average duration of a session (hours per user per day)
- DesktopGroup (name and Uid)

Example

Gets a list of all Session summary objects over 24 hours for all sessions.

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetSessionSummary?start=2011-09-30T00:00:00'&end=2011-10-01T00:00:00'

Gets a list of summary objects over 24 hours for Server OS Machine sessions

http://{dc-host}/Citrix/Monitor/OData/v1/Methods/GetSessionSummary?start=2011-09-30T00:00:00'&end=2011-10-01T00:00:00'&sessionSupportFilter=2'
Determine enum values

Jun 13, 2013

This topic explains how to determine the enum types and values returned by the Monitor Service OData API.

Several of the Monitor Service OData methods return enums, such as GetSessionCountTrend and GetMachineFailureTrend. To determine the value of these enums, use the following "metadata" OData APIs:

- **GetAllMonitoringEnums** returns an IQueryable<MonitoringEnumeration>, where MonitoringEnumeration is a class that encapsulates an enum type name and a set of values.
  
  ```csharp
  public IQueryable<MonitoringEnumeration> GetAllMonitoringEnums()
  ```

- **GetMonitoringEnum** returns the set of values for the specific type provided.
  
  ```csharp
  public IQueryable<MonitoringEnumeration> GetMonitoringEnum(string typeName)
  ```

The following enums are used as parameters in the Monitor Service OData APIs:

### sessionSupportFilter

- maps to enum named SessionSupportCode
- used in:
  - GetSessionCountTrend
  - GetSessionSummary
  - GetConnectedUsersTrend
  - GetConnectionFailureTrend
  - GetMachineFailureTrend
  - GetMachineFailureTrendsByType
- **Values:**
  - 0 = Unknown (placeholder - do not use)
  - 1 = Single Session (Desktop OS Machines)
  - 2 = Multi Session (Server OS Machines)

### connectionStateFilter

- maps to enum named ConnectionState
- used in:
  - GetSessionCountTrend
  - GetSessionSummary
- **Values:**
  - 0 = Unknown (placeholder - do not use)
  - 1 = Connected
  - 2 = Disconnected
  - 3 = Terminated
  - 4 = Preparing
  - 5 = Active
  - 6 = Reconnecting
  - 7 = Non-brokered session
  - 8 = Other
  - 9 = Pending
connectionFailureFilter

- maps to enum named ConnectionFailureType
- used in:
  - GetConnectionFailureTrendsByType
  - GetConnectionFailureTrendsByTypeLatest
  - GetConnectionFailureTrend
- Values:
  - 0 = None
  - 1 = Client Connection Failure
  - 2 = Machine Failure
  - 3 = No Capacity Available
  - 4 = No License Available
  - 5 = Configuration

machineFailureTypeFilter

- maps to enum named MachineFaultStateCode
- used in:
  - GetMachineFailureTrend
  - GetMachineFailureTrendsByType
  - GetMachineFailureTrendsByTypeLatest
- Values:
  - 0 = Unknown (placeholder - do not use)
  - 1 = None
  - 2 = Failed To Start
  - 3 = Stuck On Boot
  - 4 = Unregistered
  - 5 = Maximum Capacity
Examples

Jan 06, 2015

The following examples show how to export Monitor Service data using the OData API. This topic also provides a list of URLs for available data sets.

Example 1 - Raw XML

1. Place the URL for each data set into a web browser that is running with the appropriate administrative permissions for the XenDesktop site. Citrix recommends using the Chrome browser with the Advanced Rest Client add-in.
2. View the source.

Example 2 - PowerPivot with Excel

1. Install Microsoft Excel.
2. Follow the instructions here to install PowerPivot (depending on whether or not you are using 2010 or 2013):
3. Open Excel (running with the appropriate administrative permissions for the XenDesktop site).
   ● Using Excel 2010
   1. Click the PowerPivot tab.
   2. Click PowerPivot Window.
   3. Click From Data Feeds in the ribbon.
   4. Choose a Friendly Connection Name (for example: XenDesktop Monitoring Data) and enter the data feed url: http://{dc-host}/Citrix/Monitor/OData/v1/Data (or https: if you are using SSL).
   5. Click Next.
   6. Select the tables you want to import into Excel and click Finish. The data is retrieved.
   ● Using Excel 2013
   1. Click the Data tab.
   2. Choose From Other Sources > From OData Data Feed
   3. Enter the data feed url: http://{dc-host}/Citrix/Monitor/OData/v1/Data (or https: if you are using SSL) and click Next.
   4. Select the tables you want to import into Excel and click Next.
   5. Accept name defaults or customize names and click Finish.
   6. Choose Connection Only or Pivot Report. The data is retrieved.

You can now use PowerPivot to view and analyze the data with PivotTables and PivotCharts. For more information, see the Learning Center:

Example 3 - LinqPad

1. Download and install the latest version of LinqPad from http://www.linqpad.net.
2. Run LinqPad with the appropriate administrative permissions for the XenDesktop site.
   Tip: the easiest way is to download, install and run on the Delivery Controller.
3. Click the Add connection link.
4. Choose WCF Data Services 5.1 (OData 3) and click Next.
5. Enter the data feed URL: http://{dc-host}/Citrix/Monitor/OData/v1/Data (or https: if you are using SSL). If necessary, enter the username and password to access the Delivery Controller. Click OK.

6. You can now run LINQ queries against the data feed and export the data as needed. For example, right-click Catalogs and choose Catalogs.Take(100). This returns the first 100 Catalogs in the database. Choose Export>Export to Excel with formatting.

**URLs for Available Data Sets**

<table>
<thead>
<tr>
<th>URL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/Catalogs</td>
<td>Catalog images in the site</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/ConnectionFailureCategories</td>
<td>Grouping for connection failure types</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/ConnectionFailureLogs</td>
<td>Log of each connection failure in the site</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/Connections</td>
<td>Represents an initial connection or reconnect for a session</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/DesktopGroups</td>
<td>Delivery Groups in the site</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/FailureLogSummaries</td>
<td>Failures (connection/machine) counts by time period and Delivery Group</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/Hypervisors</td>
<td>Hosts (hypervisors) in the site</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/LoadIndexes</td>
<td>Load Index data received from the Virtual Delivery Agent (VDA)</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/LoadIndexSummaries</td>
<td>Load Index averages by time period and machine</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/MachineFailureLogs</td>
<td>Log of each machine failure by start and end date in the site</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/Machines</td>
<td>Machines in the site</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/SessionActivitySummaries</td>
<td>Session counts and logon data by time period and delivery group</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/Sessions</td>
<td>Represents a user connected to a desktop</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/TaskLogs</td>
<td>Log of all tasks and their status that have been run as part of the internal Monitoring Service</td>
</tr>
<tr>
<td>http://{dc-host}/Citrix/Monitor/OData/v1/Data/Users</td>
<td>Users that have launched a session in the site</td>
</tr>
</tbody>
</table>
Third party notices

Oct 26, 2015
XenDesktop 7.1 may include third party software licensed under the terms defined in the following documents:

- XenApp 7.0, 7.1, and 7.5 and XenDesktop 7.0, 7.1, and 7.5 Third Party Notices
- FLEXnet Publisher Documentation Supplement: Software Licenses