Migrate configuration to Citrix Cloud

Contents

Backup or Migrate your configuration

December 5, 2023

This feature helps you take a backup of your DaaS configurations. Backups ease the process of migrating the configurations from one cloud site to another. It also facilitates immediate rebounding of a site in times of emergencies.

You can use the following methods to take backups:

1. Backup + Restore

   a) Integrated with WebStudio.

2. Automated Configuration Tool (ACT)

   a) Powershell-based tool. Install the tool to use it.

The backups can be used for:

1. Restoration
2. Migration

Citrix recommends the following tools for the described scenarios.

Backup

Environment	Use Case	Recommended Tool	Special Considerations	Link
DaaS	On-demand and scheduled backups	Backup + Restore	Citrix keeps the backup and the user can download it if needed	Backup + Restore in Studio
On-premises	On-demand backups	ACT	User keeps the backup	Backup and Restore using Automated Configuration Tool

Migration

Environment	Use Case	Recommended Tool	Special Considerations	Link
On-premise to cloud	Migrate one on-premise site to DaaS	ACT		Migrate from on-premises to cloud
	Consolidate multiple on-premises sites to one DaaS site	ACT	Site merging	Merge multiple on-premises sites to a single cloud site
On-premise to on-premise	Migrate one on-premise site to another on-premise site	ACT		POC Guide:Automated Configuration Tool- On-premises to On-premises migration
	Consolidate multiple on-premises sites to another on-premise site	ACT	Site merging	POC Guide:Automated Configuration Tool On-premises to On-premises migration
				Merge multiple on-premises sites to a single cloud site
Cloud to cloud	Migrate one DaaS site to another DaaS site	ACT		Migrate from cloud to cloud
Consolidate multiple DaaS sites to one DaaS site	ACT	Site merging		Migrate from cloud to cloud

Environment	Use Case	Recommended Tool	Special Considerations	Link
				[Migrate multiple on-premises sites to a single cloud site](#)

Backup and Restore Configuration

December 23, 2023

This feature allows administrators to take a backup of their on-premises configurations. It ensures that there is always a copy of your Citrix configurations available, ready to be used. Backup along with the restore feature facilitates the migration of site information from on-premises to the cloud or from one cloud site to another with ease. If accidental changes arise, choose a backup, and select restore, and with a few clicks, you're up and running again. You can also choose to backup and restore specific components.

The feature allows you to back up and restore the following:

- Tags

- AdminRoles

- AdminScopes

- HostConnections

  > **Note:**
  >
  > You can now backup Host Connections in DaaS. Secrets must be entered each time. A pre-import check mode is performed to only ask secrets for those Host Connections that are imported or updated.

- StoreFront

- Delivery groups

- ApplicationGroups

- Applications

- AppVIsolationGroups

- AppLibPackageDiscovery

- GroupPolicies

- UserZonePreference

- Machine Catalogs

- Admin folders

    – Application folders
    – Delivery group folders
    – Machine catalog folders

- SiteSettings

Important:

Make sure that you follow the order of the list given above to restore different components. For example, if you want to restore Applications, Delivery groups, and Group policies, you must follow the order:

- Delivery groups
- Applications
- Group policies.

Note:

You cannot back up or restore MCS machine catalogs.

Type of backups

Use the following options to perform backups according to your requirements:

- **On-demand**: Backs up site configurations in real time. This action backs up the current Citrix Cloud site configurations.

- **Scheduled**: Backs up site configurations at scheduled intervals. This recurs at the intervals mentioned.

Note:

Only administrators with full rights and cloud administrators can do on-demand or scheduled backups and restoration.

Backup and Restore

January 2, 2024

This article describes the actions that you can perform on the **Backup and Restore** blade available in the left pane of your Manage DaaS console.

Backup and Restore allows the administrator to take backups from the Studio UI.

To create on-demand backups

1. Sign in to Citrix Cloud.
2. In the **DaaS** tile, click **Manage**.
3. From the left-menu, select **Backup and Restore**.
4. Within the **Backup and Restore** tab, click **Backup** button.
5. On the **Backup** window, enter the notes, and select **Backup**.

After the backup completes successfully. It is listed as a line item under the **Backup** button in the **Backup** tab.

1. Select a particular backup to see more information on the backup in the **Details** tab.
2. You can pin or delete a selected backup using the pin or delete button respectively. You can perform the same action by right-clicking the selected backup.

> **Note:**
>
> Citrix stores the last 30 backups. When you create more than 30 backups, the system deletes old backups in the order they were created. For every new backup that is created in addition to the 30 backup limit, the system deletes the backup that was created first. **Pin Backup** is used to prevent automatic deletion of a particular backup. You can pin up to 5 backups. In case there are pinned backups, we retain all pinned backups in addition to 30 backups that is at most, you can retain 35 backups –30 non-pinned, and 5 pinned backups.

To create scheduled backups

1. Sign in to Citrix Cloud.
2. In the **DaaS** tile, click **Manage**.
3. From the left menu, select **Backup + Restore**.
4. Go to the schedule tab and select **Create Schedule**.
5. On the **Create Backup** window, enter the following:

- **Enable this backup**. Choose **yes** to enable this backup schedule.
- **Name**. Enter a name for the backup schedule.
- **Description**. Enter a description.
- **Time zone**. Select the time zone of your choice.
- **Frequency**. Select the frequency to run the backup.
- **Repeats every**. Enter the number of days after which the schedule must be repeated.
- **Start date**. Enter the start date.
- **Begin backup at**. Enter the time when you want the backup to begin in a 24-hour format.

6. Select **Done**.

7. Select **View Restore Details** to see more information about the restore.

Restore

This feature allows the administrators to restore the site configurations from the backups. We allow you to restore one component at a time.

> **Note:**
>
> Restoring only one component at a time ensures that there are no conflicts during restoration due to the dependencies between the components.

1. Sign in to Citrix Cloud.

2. In the **DaaS** tile, click **Manage**.

3. From the left menu, select **Backup and Restore**.

4. Click a specific backup from the list.

5. Click **Restore Backup** button.

6. On the **Restore Backup** window, enter the following:

   - **Components**. Select the components that you want to restore.

   - **Filters**. Specify the member names that you want to restore in a comma-separated list. For example, filtering for the word *chrome* restores only objects with the name chrome. Similarly, the phrase *chr*** restores objects beginning with chr.

     > **Note:**
     >
     > Select a backup and navigate to **Details > Components > View Members** to define filters corresponding to existing components and their members.

   - **Check mode**. Choose **yes** to confirm what would be restored when an actual restore operation is done. This option pre-evaluates everything needed for a restore without actually

triggering one. For example, it displays the list of applications that will be restored when the restore process is run for the applications component.

> **Note:**
>
> Check mode does not run the actual restore. This option also does not check for any dependencies between the components. For optimal results, we recommend using the check mode on one component at a time.

- **Restore type**. Choose either one:
    - **Missing only**. Use this restore type to add components and members to a site. This option does not change the existing components and members even if the backup being restored has different configurations.
    - **Missing and updating existing**. Use this restore type to add and update components to a site. This option updates the existing components and members with the configurations of the backup if both have different configurations.
- **Notes**. Enter any notes.

7. Select **Restore**.

The restore results are listed in the Restore window after the restore action is completed. All restore actions are listed under the **History** tab.

History

You can track all the **Backup+Restore** actions in the **History**. This page gives you the following information:

- Action performed
- Components and members backed up
- Who performed the activity

Use the search bar on the top right in the **History** tab to search for a specific backup. For example: you can search for backups taken by a particular admin or taken on a particular day.

Backup and Restore using the Automated Configuration tool

December 5, 2023

Automated Configuration tool allows you to take on-demand backups in the following environments:

- Citrix Virtual Apps and Desktops
- DaaS

Citrix Virtual Apps and Desktops

Prerequisites for backing up your configurations

- The current release or the immediate predecessor of Citrix Virtual Apps and Desktops or all versions of Citrix Virtual Apps and Desktops, XenApp and XenDesktop LTSRs.

- A machine to run Auto Config. This device can be either:

  - A delivery controller
  - A domain joined machine with .NET Framework 4.7.2 or later and the Citrix PowerShell SDK. For Citrix PowerShell, you must install Citrix Studio.

Back up

1. Install Automated Configuration. For more information, see Download Automated Configuration.
2. Double-click **Auto-Config**.
3. On the PowerShell window, run the command `Backup-CvadAcToFile` to export all components. Exporting your on-premises configuration does not modify it.

After you run the command for the first time, the process creates a folder with the .yml configuration files at the following location:
`%HOMEPATH%\Documents\Citrix\AutoConfig`.

The folder contains the backup files from the recent export, and each previous backup is stored in a subfolder.

Restore

1. Double-click **Auto-Config**.
2. Run the following command to do the restore:
   ```
   Restore-CvadAcToSite -Environment OnPrem -RestoreFolder <folder
   path of the backup files>.
   ```

After you run the cmdlet, verify the state of the export, and the log folder is available at:
`%HOMEPATH%\Documents\Citrix\AutoConfig`.

If there are any errors or exceptions, see the **Fixups** section in the log file.

DaaS

Prerequisites for backing up your configurations

- A DaaS environment with an active resource location and Connector installed.
- Connectivity to sites accessing Citrix Cloud. For more information, see System and Connectivity Requirements.

> **Note**
>
> You cannot backup MCS from the cloud using Automated Configuration.

Back up

- CustomerInfo.yml file with your customer ID, client ID, and secret key information. For more information on the process to retrieve your customer ID, client ID, and secret key, see Generating the customer ID, client ID, and secret key. For information on how to add this information to the CustomerInfo.yml file, see Populating customer info file.
- ZoneMapping.yml file with information that maps your resource locations in the cloud. For more information on how to map your zones, see the Populating zone mapping file.
- CvadAcSecurity.yml file with information on host connections.

1. Install Automated Configuration. For more information, see Install Automated Configuration.

> **Note**
>
> For DaaS backups, install the Automated Configuration on a machine with Internet access and to which the administrator has direct access as well.

1. Double-click **Auto Config**.
2. On the PowerShell window, run the following command to take a backup:
   `Export-CvadAcToFile`

After you run the command for the first time, the process creates a folder with the .yml configuration files at the following location:
`%HOMEPATH%\Documents\Citrix\AutoConfig`

The folder contains the backup files from the recent export, and each previous backup is stored in a subfolder.

Restore

1. Double-click the **Auto Config**.

2. On the PowerShell window, run the following command to do the restore:

```
Restore-CvadAcToSite -RestoreFolder <folder path of the backup
files><!--NeedCopy-->
```

After you run the cmdlet, verify the state of the export, and you can locate the log folder at `%HOMEPATH%\Documents\Citrix\AutoConfig`.

If there are any errors or exceptions, see the **Fixups** section in the log file.

Granular migration

> **Note**
>
> For more information on the component migration order, see Component Migration Order.

Restore entire components

To restore a component, you must select one or more parameters.
For example, to restore the entire delivery group and machine catalog components, run the following command:

```
Restore-CvadAcToSite -RestoreFolder %HOMEPATH%\Documents\Citrix\
AutoConfig/Backup_yyyy_mm_dd_hh_mm_ss -DeliveryGroups -MachineCatalogs
```

Restore component members

To restore one or more component members, use the `IncludeByName` feature. You must invoke the `Restore` cmdlet with the `Restorefolder` parameter along with the selected component, and the inclusion list.

For example, to restore two group policies from a backup:

```
Restore-CvadAcToSite -RestoreFolder %HOMEPATH%\Documents\Citrix\
AutoConfig/Backup_yyyy_mm_dd_hh_mm_ss -GroupPolicies -IncludeByName
Policy1,Policy2
```

Restore complete cloud site configuration

To restore all the components or the entire cloud site, run the following command:

```
Restore-CvadAcToSite -RestoreFolder %HOMEPATH%\Documents\Citrix\
AutoConfig/Backup_yyyy_mm_dd_hh_mm_ss
```

Migrate configuration to Citrix Cloud

December 5, 2023

Why use Automated Configuration

IT administrators in charge of large or complex environments often find migrations to be a tedious process. They frequently end up writing their own tools to accomplish this task successfully since it tends to be specific to their use cases.

Citrix wants to help ease this process by automating the migration process using the Automated Configuration tool. Administrators can easily test current configurations in Citrix Cloud and take advantage of the benefits offered by Citrix DaaS (formerly Citrix Virtual Apps and Desktops service) while keeping their current environments *intact*. There is also no end user impact, as Automated Configuration works seamlessly in the background. Such benefits include reduced administrative overload when Citrix manages part of the back-end and control plane, automatic and customizable Citrix Cloud component updates, and others.

Citrix uses industry-standard configuration as code to provide a mechanism to help automate migration processes. Automated Configuration discovers and exports one or more on-premises sites as a collection of configuration files. These files'configuration can then be imported into Citrix DaaS.

Automated Configuration also allows administrators to merge multiple on-premises sites into a single site, while avoiding name collisions. Administrators can control whether the on-premises or cloud configuration controls resources.

Automated Configuration is not just a one time migration tool, but can also automate your day-to-day configuration in Citrix Cloud. Moving your Citrix DaaS configuration can be beneficial for many reasons:

- Syncing your site from test or stage to production
- Backing up and restoring your configuration
- Reaching resource limits
- Migrating from one region to another

The following *2-minute* video provides a quick tour of Automated Configuration.

This is an embedded video. Click the link to watch the video

For additional information on Automated Configuration, see Proof of Concept: Automated Configuration Tool on Tech Zone.

For a deeper look into moving your deployment and readying your on-premises configuration for migration, see Deployment Guide: Migrating Citrix Virtual Apps and Desktops from on-premises to Citrix Cloud on Tech Zone.

Download Automated Configuration

Download and install the Automated Configuration tool from Citrix Downloads.

> **Important:**
>
> To prevent errors in functionality, always use the latest available version of Automated Configuration.

Upgrading Automated Configuration

When running cmdlets that access the cloud in Automated Configuration, the tool alerts you when there is a newer version available for download.

You can make sure you have the latest version by following the steps below:

1. Double-click the **Auto Config** icon. A PowerShell window appears.

2. Run the following command to check your version number.

   ```
   Get-CvadAcStatus
   ```

3. Check your tool version against the version listed in the alert or at Citrix Downloads. The latest version of the tool is located there.

4. Download and install the latest version of the tool. You do *not* need to uninstall the old version to upgrade Automated Configuration.

> **Note:**
>
> The alert appears every time you run a cmdlet that accesses the cloud. For more information on cmdlets, see Automated Configuration tool cmdlets.

Known limitations

- Machine catalogs provisioned through Machine Creation Services have special considerations. For more information on MCS, see Understanding migrating Machine Creation Services provisioned catalogs.

Supported migration objects

Automated Configuration supports moving the configuration of the following components:

- Tags
- Delegated Admin

  – Scopes
  – Roles

- Host Connections

  – A Single Resource Pool
  – Admin Scopes

- Machine Catalogs

  – Admin Scopes
  – Machines
  – Remote PC Access, Physical, Pooled, Provisioned, MCS, Assigned

- StoreFronts
- Delivery Groups

  – Access Policy
  – Admin Scope Association
  – Application Access Policy
  – Assignment Policy
  – Entitlement/Desktop Policy
  – Power Schedules
  – Session Lingering
  – Session Prelaunch
  – Reboot Schedules
  – Tags

- Application Groups

  – Admin Scope Association
  – Delivery Groups
  – Users and Groups

- Applications

  – Application Folders
  – Icons
  – Applications

- **–** Broker Configured FTAs
- **–** Tags

- Group Policies
- User Zone Preferences

Component migration order

The components and their dependencies are listed here. A component's dependencies must be in place before it can be imported or merged. If a dependency is missing, it can cause the import or merge command to fail. The **Fixups** section of the log file shows missing dependencies if an import or merge fails.

1. Tags

    - No pre-dependencies

2. Delegated Admin

    - No pre-dependencies

3. Host Connections

    - Security Information in CvadAcSecurity.yml

4. Machine Catalogs

    - Machines present in Active Directory
    - Host Connections
    - Tags

5. StoreFronts
6. Delivery Groups

    - Machines present in Active Directory
    - Users present in Active Directory
    - Machine Catalogs
    - Tags

7. Application Groups

    - Delivery Groups
    - Tags

8. Applications

    - Delivery Groups
    - Application Groups

- Tags

9. Group Policies

    - Delivery Groups
    - Tags

10. User Zone Preferences

Common prerequisites

The following are some common prerequisites that are needed for Automated Configuration to work correctly. These prerequisites are used in both on-premises to cloud and cloud to cloud migrations.

Generating the customer ID, client ID, and secret key

Before you begin your migration using Automated Configuration, you need your Citrix Cloud customer ID and you must create a client ID and a secret key to import your configuration to Citrix Cloud. All cmdlets accessing the cloud require these values.

The following steps allow you to retrieve the customer ID and create the client ID and secret key.

To retrieve the **Customer ID**:

1. Sign into your Citrix Cloud account and select the customer.

2. Click the hamburger menu, then select **Identity and Access Management** in the drop-down menu.

3. The **Customer ID** is located on the **Identity and Access Management** page.

To retrieve the **Client ID** and **Secret Key**:

1. On the **Identity and Access Management** page, click the **API Access** tab.

2. Enter a name in the box. This name is used to differentiate between multiple client IDs and secret keys. Click **Create Client** to create the client ID and the secret key.

3. The following dialog appears after you successfully create the client ID and the secret key. Be sure to copy both values to a secure location and download the .csv file containing this information. The .csv file can be used to create the CustomerInfo.yml file.

4. The client ID and the secret key are successfully created.

Place these values in a secure location and share only with trusted company members who need access to the tool or access the cloud Rest APIs. The client ID and secret key do not expire. If they are compromised, immediately remove them by using the **Trash** icon and create new ones.

> **Note:**
>
> The secret key cannot be retrieved if it is lost or forgotten; a new client ID and secret key must be created.

Populating customer info file

Using the CustomerInfo.yml file eliminates the need to provide customer information parameters with each cmdlet's execution. Any of the customer information can be overridden by using cmdlet parameters.

Create the CustomerInfo.yml file by using the `New-CvadAcCustomerInfoFile` cmdlet.

> **Important:**
>
> Do not manually edit the CustomerInfo.yml file. Doing so can cause inadvertent formatting errors.

`New-CvadAcCustomerInfoFile` has the following required parameters.

- CustomerId –customer's ID.
- ClientId –customer's client ID created on Citrix Cloud.
- Secret –customer's secret created on Citrix Cloud.

`New-CvadAcCustomerInfoFile -CustomerId markhof123 -ClientId 6813EEA6 -46CC-4F8A-BC71-539F2DAC5984 -Secret TwBLaaaaaaaaaaaaaaaaaaw==`

You can also create the CustomerInfo.yml using the `SecurityCsvFileSpec` parameter that points to the downloaded security.csv file. You must also specify the CustomerId.

`New-CvadAcCustomerInfoFile -SecurityCsvFileSpec C:\Users\my_user_name \downloads/security.csv -CustomerId markhof123`

Update the CustomerInfo.yml file by using the `Set-CvadAcCustomerInfoFile` cmdlet. This cmdlet only changes the Client ID.

`Set-CvadAcCustomerInfoFile -ClientId C80487EE-7113-49F8-85DD-2CFE30CC398E`

The following is a sample CustomerInfo.yml file.

```
1              # Created/Updated on 2020/01/29 16:46:47
2              CustomerId: ' markhof123 '
3              ClientId: ' 6713FEA6-46CC-4F8A-BC71-539F2DDK5384 '
4              Secret: ' TwBLaaabbbaaaaaaaaaw== '
5              Environment: Production
6              AltRootUrl: ' '
7              StopOnError: False
```

```
 8              AlternateFolder: ' '
 9              Locale: ' en-us '
10              Editor: ' C:\Program Files\Notepad++\notepad++.exe '
11              Confirm: True
12              DisplayLog: True
```

Populating zone mapping file

An on-premises zone is the equivalent of the cloud resource location. Unlike other site components, you cannot import the on-premises zone to the cloud automatically. Instead, it must be manually mapped using the ZoneMapping.yml file. Import failures can occur if the zone name is not associated with an existing resource location name.

For on-premises sites having only one zone and cloud sites only one resource location, the Automated Configuration tool makes the correct association, eliminating the need to manually manage the ZoneMapping.yml file.

For on-premises sites having multiple zones or cloud sites having multiple resource locations, the ZoneMapping.yml file must be manually updated to reflect the correct mapping of on-premises zones to cloud resource locations. This must be done before attempting any import operation to the cloud.

The ZoneMapping.yml file is located in *%HOMEPATH%\Documents\Citrix\AutoConfig*. The content of the .yml file is a dictionary with the zone name as the key and the resource location name as the value.

As an example, an on-premises Citrix Virtual Apps and Desktops site with a primary zone called "Zone-1"and a secondary zone called "Zone-2"is migrated to a Citrix DaaS deployment with two newly created cloud resource locations called "Cloud-RL-1"and "Cloud-RL-2". In this instance, the ZoneMapping.yml would be configured as follows:

```
1              Zone-1: Cloud-RL-1
2
3              Zone-2: Cloud-RL-2
```

> **Note:**
>
> A space must be between the colon and resource location name. If spaces are used in the zone or resource location name, enclose the name with quotes.

Host connections

Host connections and their associated hypervisors can be exported and imported using Automated Configuration.

Adding a hypervisor to a host connection requires security information specific to the type of hypervisor. This information cannot be exported from the on-premises site for security considerations. You must manually provide the information so that Automated Configuration can successfully import host connections and hypervisors to the cloud site.

The export process creates the CvadAcSecurity.yml file in *%HOMEPATH%\Documents\Citrix\AutoConfig* containing placeholders for each security item needed for the specific hypervisor type. You must update the CvadAcSecurity.yml file before importing into the cloud site. Administrator updates are retained over multiple exports with new security placeholders added as needed. Security items are never removed. For more information, see Manually update the CvadAcSecurity.yml file

```
1              HostConn1:
2              ConnectionType: XenServer
3              UserName: root
4              PasswordKey: rootPassword
5              HostCon2:
6              ConnectionType: AWS
7              ApiKey: 78AB6083-EF60-4D26-B2L5-BZ35X00DA5CH
8              SecretKey: TwBLaaaaaaaaaaaaaaaaaaw==
9              Region: East
```

Per-hypervisor security information   The following lists the security information required for each hypervisor type.

- XenServer, Hyper-V, VMware

    – User Name
    – Clear-text Password

- Microsoft Azure

    – Subscription ID
    – Application ID
    – Application Secret

- Amazon Web Services

    – Service Account ID
    – Application Secret
    – Region

Special security considerations   All security information is entered as clear text. If clear text is not recommended, the host connections and associated hypervisors can be manually created using the **Manage > Full Configuration** interface. The host connections and hypervisor names must match their on-premises counterparts exactly so that machine catalogs that use the host connections can be successfully imported.

Activating sites

The delivery controller in both on-premises and cloud sites control resources such as brokering desktops, applications, and rebooting machines. Problems occur when a common set of resources is controlled by two or more sites. Such a situation can occur when migrating from an on-premises site to a cloud site. It is possible for both the on-premises and cloud delivery controllers to manage the same set of resources. Such dual management can lead to resources becoming unavailable and unmanageable, and can be difficult to diagnose.

Site activation allows you to control where the active site is controlled.

Site activation is managed using the delivery group maintenance mode. Delivery groups are placed in maintenance mode when the site is inactive. Maintenance mode is removed from delivery groups for sites that are active.

Site activation does not affect or manage VDA registration or machine catalogs.

- `Set-CvadAcSiteActiveStateCloud`
- `Set-CvadAcSiteActiveStateOnPrem`

All cmdlets support the `IncludeByName` and `ExcludeByName` filtering. This parameter allows you to select which delivery groups can have their maintenance mode changed. Delivery groups can be selectively changed as needed.

Import and transferring control to the cloud

The following is a high-level description on how to import and transfer control from the on-prem site to the cloud site.

1. Export and import the on-premises site to the cloud. Make sure the `-SiteActive` parameter is not present on any of the import cmdlets. The on-premises site is active and the cloud site inactive. By default, cloud site delivery groups are in maintenance mode.
2. Verify the cloud content and configuration.
3. During off hours, set the on-premises site to inactive. The `-SiteActive` parameter must be absent. All on-premises site delivery groups are in maintenance mode.

   - `Set-CvadAcSiteActiveStateOnPrem`

4. Set the cloud site to active. The `-SiteActive` parameter must be present. No cloud site delivery groups are in maintenance mode.

   - `Set-CvadAcSiteActiveStateCloud -SiteActive`

5. Verify that the cloud site is active and the on-premises site is inactive.

Transferring control back to the on-premises site

To transfer control from the cloud site to the on-premises site:

1. During off hours, set the cloud site to inactive. All cloud site delivery groups are in maintenance mode.

    - `Set-CvadAcSiteActiveStateCloud`

2. Set the on-premises site to active. No on-prem site delivery groups are in maintenance mode.

    - `Set-CvadAcSiteActiveStateOnPrem -SiteActive`

Additional site activation information

- If no machines are power managed and there are no reboot schedules (which usually means there are no host connections either) all cloud delivery groups can be imported as active. Add `-SiteActive` to `Merge-CvadAcToSite`/`Import-CvadAcToSite` or run `Set-CvadAcSiteActiveStateCloud -SiteActive` after importing.
- If machines are power managed or there are reboot schedules, a different process is needed. For example, when switching from on-premises to cloud in this situation, set the on-premises site to inactive using `Set-CvadAcSiteActiveStateOnPrem`. Then, set the cloud site to active using `Set-CvadAcSiteActiveStateCloud -SiteActive`.
- The `Set-CvadAcSiteActiveStateCloud` and `Set-CvadAcSiteActiveStateOnPrem` cmdlets are also used to reverse the process. For example, run `Set-CvadAcSiteActiveStateCloud` without the `-SiteActive` parameter, then run `Set-CvadAcSiteActiveStateOnPrem` with the `-SiteActive` parameter.

Understanding migrating Machine Creation Services provisioned catalogs

> **Note:**
>
> This feature is available only on versions 3.0 and later. Check your version by using `Get-CvadAcStatus` within Automated Configuration.

Machine Creation Services (MCS) catalogs create two different types of catalogs:

- When changes made to a machine are lost/reversed (Commonly Server OS, where applications are published) —this is a pooled VDI / multi-session use case
- When changes made to a machine are preserved across reboot (Commonly client OS with a dedicated user) —this is a static VDI use case

The type of catalog can be confirmed in the catalog node in Citrix Studio and looking at the "User data:"value of the catalog.

> **Note:**
>
> MCS cannot be backed up from the cloud using Automated Configuration.

Pooled VDI / multi-session catalogs

Catalogs with "User data: Discard"are pooled VDI catalogs and can only migrate the main image and configuration. Any virtual machines in these catalogs are not migrated. This is because the life cycle of the virtual machine is maintained by the site you are importing from, which means every time the machines are turned on, its state might change. This makes import impossible as the import data for the virtual machines quickly gets of out of sync.

When you are migrating these catalogs using the tool, it creates catalog metadata and initiates main image creation, but no machines are imported.

Since this process can take some time to be created based on the size of the main image, the import command within the tool only starts the MCS catalog creation and does not wait for it to finish. After the import has completed, monitor the catalog create progress using the Full Configuration management interface in the cloud deployment.

Once the main image is created, you can provision machines. Capacity considerations need to be taken into account since you would have capacity consumed from your on-premises usage.

All other objects (delivery groups/applications/policies, and so forth) that use that catalog can be imported and do not have to wait for the main image creation. When the catalog has finished creating, machines can be added to the imported catalog and then users can launch their resources.

> **Note:**
>
> Use the same commands available within the tool to migrate catalogs and all other objects.

Static VDI catalogs

> **Note:**
>
> Since this operation imports low-level details that are stored in the database, this process must be run from a machine with database access.

static VDI catalogs migrate the main image, configurations, and all virtual machines. Unlike the pooled VDI use case, no images need to be created.

The VDAs must be pointed to the connector for them to register with the cloud.

Refer to the Activating sites section to make the cloud site active, so that the reboot schedule, power management, and other items are controlled by the cloud.

Once the migration is completed, if you want to delete this catalog from your on-premises site, you must select leave VM and AD account. Otherwise, they are deleted and the cloud site would be left pointing to the deleted VM.

Migrating from on-premises to cloud

December 5, 2023

Automated Configuration allows you to automate moving your on-premises configuration to a cloud site.

The following image is a high-level view of what Automated Configuration can do to migrate your configuration to the cloud.

Prerequisites for migrating your configuration

For *exporting* your configuration from Citrix Virtual Apps and Desktops, you need:

- Citrix Virtual Apps and Desktops: current release and its immediate predecessor or Citrix Virtual Apps and Desktops, XenApp and XenDesktop LTSRs: all versions
- A domain-joined machine with .NET Framework 4.7.2 or later and the Citrix PowerShell SDK. This is automatically installed on the Delivery Controller. (To run on machine other than the on-premises Delivery Controller, Citrix Studio must be installed, as Studio installs the correct PowerShell snap-ins. The Studio installer can be found on the Citrix Virtual Apps and Desktops installation media.)

For *importing* your configuration into Citrix DaaS (formerly Citrix Virtual Apps and Desktops service), you need:

- A machine with access to Citrix Cloud. This does not have to be a Delivery Controller or a domain-joined machine.
- Citrix DaaS provisioned.
- An active resource location with Connector installed and domain-joined to the same domain as the on-premises setup.
- Connectivity to sites accessing Citrix Cloud must be allowed and available. For more information, see System and Connectivity Requirements.

> **Note:**
>
> Automated Configuration cannot be installed on a Cloud Connector system.

Exporting your Citrix Virtual Apps and Desktops on‑premises configuration

Important:

- You must have your CustomerInfo.yml file with your customer ID, client ID, and the secret key information included. For more information on how to retrieve your customer ID, client ID and secret key, see Generating the customer ID, client ID, and secret key. For information on how to add this information to the CustomerInfo.yml file, see Populating customer info file.
- The ZoneMapping.yml file must include information that maps your on‑premises zone to Resource Locations in the cloud. For more information on how to map your zones, see Populating zone mapping file.
- If you have host connections, you must input the corresponding info in the CvadAcSecurity.yml file.

1. Install Automated Configuration.

2. Double‑click the **Auto Config** icon. A PowerShell window appears.

3. Run the following command to export all components. Exporting your on‑premises configuration does *not* change it in any way.

   ```
   Export-CvadAcToFile
   ```

After you run any cmdlet for the first time, an export folder with the .yml configuration files and logs is created. The folder is at *%HOMEPATH%\Documents\Citrix\AutoConfig*. Each successive export creates a subfolder. The parent folder *%HOMEPATH%\Documents\Citrix\AutoConfig* always contains the exported files from the most recent export.

Note:

If Automated Configuration is not installed on the Delivery Controller, run **import**-module **Citrix.AutoConfig.Commands** before using the tool through PowerShell. This step is not needed if you open Automated Configuration using the **Auto Config** icon.

If you encounter any errors or exceptions, see the **Fixups** section in the log file.

Importing your configuration to Citrix DaaS

Important:

- You must have your CustomerInfo.yml file with your customer ID, client ID, and the secret key information included. For more information on how to retrieve your customer ID, client ID and secret key, see Generating the customer ID, client ID, and secret key. For information on how to add this information to the CustomerInfo.yml file, see Populating customer info

file.

- The ZoneMapping.yml file must include information that maps your on-premises zone to resource locations in the cloud. For more information on how to map your zones, see Populating zone mapping file.
- If you have host connections, you must input the corresponding info in the CvadAcSecurity.yml file.

Running an import

1. Double-click the **Auto Config** icon. A PowerShell window appears.

2. Run the following command to import all components.

   `Merge-CvadAcToSite`

Verify the expected state with the new current state. Various import options control whether the import results are identical or a subset of the on-premises site.

After you run the cmdlet, an export folder with the .yml configuration files and logs is created. The folder is at *%HOMEPATH%\Documents\Citrix\AutoConfig*.

If you encounter any errors or exceptions, see the **Fixups** section in the log file.

> **Note:**
>
> If Automated Configuration is not installed on the Delivery Controller, run **import**-module `Citrix.AutoConfig.Commands` before using the tool through PowerShell. This step is not needed if you open Automated Configuration using the **Auto Config** icon.

To revert to your original Citrix DaaS configuration, see Backing up your Citrix DaaS configuration.

Import operation in detail

The import process is designed to accurately perform updates, only perform needed updates and verify that all updates have been correctly made. The steps followed in all import operations follow.

1. Read the exported .yml file (expected state).
2. Read the cloud (current state).
3. Back up the pre-import cloud state to .yml files (pre-backup can be restored if necessary).
4. Evaluate the differences between the expected and current state. This determines which updates to make.
5. Make the updates.
6. Reread the cloud (new current state).
7. Back up the post-import cloud state to .yml files (post-backup can be restored if necessary).

---

8. Compare the new current state with the expected state.

9. Report the results of the comparison.

Granular migration

> **Important:**
>
> For more information on component migration order, see Component migration order.

You can selectively migrate components only or even component names only.

- Component parameters supported include `MachineCatalogs`, `Tags` and more.
- Component name parameters supported include `IncludeByName` and `ExcludeByName` parameters, and others.

For more information on parameters and how to use them, see Granular migration parameters.

Activating sites

Site activation allows you to control which site is active and controls your resources. For more information, see Activating sites.

Merging multiple sites into a single site

December 5, 2023

Multi-site support for Automated Configuration provides a method to merge multiple on-premises sites into a single cloud site.

Multi-site support adds unique prefixes and suffixes to component names on a per on-premises-site basis, ensuring name uniqueness after multiple on-premises sites are merged to a single cloud site.

Prefixes and suffixes can be assigned for each of the following components on a per-on-premises-site basis.

- `AdminScope`
- `AdminRole`
- `ApplicationAdmin`
- `ApplicationFolder`
- `ApplicationGroup`
- `ApplicationUser`

- `DeliveryGroup`
- `GroupPolicy`
- `HostConnection`
- `MachineCatalog`
- `StoreFront`
- `Tag`

Application folders support prefixing, suffixing, and rerooting. Rerooting adds an extra top level folder to an application's existing folder structure.

Prefixing and suffixing rules

1. Prefixes and suffixes cannot contain any of the following special characters: `\ , / ; : # . * ? = < > | ( )"' { } [ ]`
2. Prefixes and suffixes can contain trailing spaces but not leading spaces.
3. Prefixes and suffixes must be double quoted to contain trailing spaces.
4. Prefixes and suffixes are applied at the time of import, merge, and add. The source .yml files are never modified.
5. The prefix and suffix process automatically prefixes or suffixes dependent component names when applicable. For example, if machine catalog names are prefixed with "East,"delivery groups referencing them are also prefixed with "East."
6. If a component name already begins with the prefix or suffix, no prefix or suffix is added. Component names cannot contain double identical prefixes or suffixes.
7. Prefixes and suffixes can be individually used or used in combination.
8. Use of a prefix or a suffix on a component is optional.

> **Note:**
>
> The Full Configuration interface displays components in alphabetical order.

Group by site

Use prefixing to visually group components from a single site. Each site is listed in its own group with prefixing alphabetically controlling the ordering of different site groups.

Group by name

Use suffixing to visually group like-named components from multiple sites. Like-named components from different sites visually alternate.

SitePrefixes.yml file

Site prefixing begins with the SiteMerging.yml file that contains the site prefix and suffix mapping for one or more on-premises sites. You can manage the SiteMerging.yml file manually, or by using the available cmdlets listed at the Merging multiple on-premises sites cmdlets section.

Exporting, importing, merging, and adding

Merging cannot begin until you have exported an on-premises site. To export an on-premises site, see Migrating from on-premises to cloud.

Central export target folder

The methods described in this section place multiple site exports into a central file share location. The SiteMerging.yml file, CustomerInfo.yml file, and all export files reside in that file share location, allowing you to do the import from one location independent of the on-premises sites.

Cloud accessing operations never reference the on-premises sites or Active Directory, therefore allowing you to do cloud-accessing operations from anywhere.

Direct file share

The export, import, merge, and new/add operations provide a parameter to target or source a folder other than the default folder, *%HOMEPATH%\Documents\Citrix\AutoConfig*. The following examples use a central file share located at `\\share.central.net` that the admin already has access to, having provided credentials as needed.

To target the export to a site-specific folder, use the `-TargetFolder` parameter:

From the East DDC:

```
mkdir \\share.central.net\AutoConfig\SiteEast
```

```
Export-CvadAcToFile -TargetFolder \\share.central.net\AutoConfig\SiteEast
```

From the West DDC:

```
mkdir \\share.central.net\AutoConfig\SiteWest
```

```
Export-CvadAcToFile -TargetFolder \\share.central.net\AutoConfig\SiteWest
```

After the exports are complete, create the CustomerInfo.yml and SiteMerging.yml files and place them in `\\share.central.net\AutoConfig`.

> **Note:**
>
> Do not use the `SiteRootFolder` parameter when creating the SitePrefixes.yml when using this direct file share reference method.

To import, merge, or add from the direct file share, you must decide from which machine you want to do the cloud accessing operation. Options include:

- One of the on-premises DDCs where the tool is already installed.
- The machine hosting the file share.
- A different machine.

Automated Configuration must be installed on the machine accessing the cloud. Neither the on-premises PowerShell SDK, DDC, nor Active Directory are used, so the cloud accessing execution requirements are simpler than the export requirements.

To merge the East DDC to the cloud:

```
Merge-CvadAcToSite –SiteName East –SourceFolder \\share.central.
net\AutoConfig\SiteEast -CustomerInfoFileSpec \\share.central.net\
AutoConfig\CustomerInfo.yml
```

To merge the West DDC to the cloud:

```
Merge-CvadAcToSite –SiteName West –SourceFolder \\share.central.
net\AutoConfig\SiteWest -CustomerInfoFileSpec \\share.central.net\
AutoConfig\CustomerInfo.yml
```

The following is a sample SitePrefixes.yml file used in the previous example.

```
 1    East:
 2      SiteRootFolder: "" # Important: leave this empty
 3      AdminScopePrefix: "East_"
 4      AdminRolePrefix: "East_"
 5      ApplicationAdminPrefix: "East_"
 6      ApplicationFolderPrefix: "" # Note that a new parent root folder
             is used instead
 7      ApplicationFolderRoot: "East"
 8      ApplicationGroupPrefix: "East_"
 9      ApplicationUserPrefix: "East_"
10      DeliveryGroupPrefix: "East_"
11      GroupPolicyPrefix: "East_"
12      HostConnectionPrefix: "East_"
13      MachineCatalogPrefix: "East_"
14      StoreFrontPrefix: "East_"
15      TagPrefix: "East_"
16      AdminScopeSuffix: "_east"
17      AdminRoleSuffix: "_east"
18      ApplicationAdminSuffix: "_east"
19      ApplicationFolderSuffix: "_east"
```

```
20          ApplicationGroupSuffix: "_east"
21          ApplicationUserSuffix: "_east"
22          DeliveryGroupSuffix: "_east"
23          GroupPolicySuffix: "_east"
24          HostConnectionSuffix: "_east"
25          MachineCatalogSuffix: "_east"
26          StoreFrontSuffix: "_east"
27          TagSuffix: "_east"
28        West:
29          SiteRootFolder: "" # Important: leave this empty
30          AdminScopePrefix: "Western "
31          AdminRolePrefix: "Western "
32          ApplicationAdminPrefix: "Western "
33          ApplicationFolderPrefix: "" # Note that a new parent root folder
                is used instead
34          ApplicationFolderRoot: "Western"
35          ApplicationGroupPrefix: "Western "
36          ApplicationUserPrefix: "Western "
37          DeliveryGroupPrefix: "Western "
38          GroupPolicyPrefix: "Western "
39          HostConnectionPrefix: "Western "
40          MachineCatalogPrefix: "Western "
41          StoreFrontPrefix: "Western "
42          TagPrefix: "Western "
43          AdminScopeSuffix: ""
44          AdminRoleSuffix: ""
45          ApplicationAdminSuffix: ""
46          ApplicationFolderSuffix: ""
47          ApplicationGroupSuffix: ""
48          ApplicationUserSuffix: ""
49          DeliveryGroupSuffix: ""
50          GroupPolicySuffix: ""
51          HostConnectionSuffix: ""
52          MachineCatalogSuffix: ""
53          StoreFrontSuffix: ""
54          TagSuffix: ""
```

File share reference using SiteMerging.yml

This method uses the `SiteRootFolder` member of the site's prefixes set. While more involved than the direct file share method, this method reduces the odds of targeting the wrong folder when exporting, importing, merging, or adding.

First, set the `SiteRootFolder` for each site in the SiteMerging.yml file. You must do this on the shared location.

```
New-CvadAcSiteMergingInfo –SiteName East –SiteRootFolder \\share.
central.net\AutoConfig\SiteEast -SitePrefixesFolder \\share.central.
net\AutoConfig
```

```
New-CvadAcSiteMergingInfo –SiteName West –SiteRootFolder SiteWest -
SitePrefixesFolder \\share.central.net\AutoConfig
```

In this example, East is a fully qualified folder specification and West is a relative folder specification.

To target the export to a site-specific folder using the SiteMerging.yml file:

From the East DDC:

```
mkdir \\share.central.net\AutoConfig\SiteEast
```

```
Export-CvadAcToFile -SiteName East -CustomerInfoFileSpec \\share.
central.net\AutoConfig\CustomerInfo.yml
```

From the West DDC:

```
mkdir \\share.central.net\AutoConfig\SiteWest
```

```
Export-CvadAcToFile -SiteName West -CustomerInfoFileSpec \\share.
central.net\AutoConfig\CustomerInfo.yml
```

The export cmdlet uses the CustomerInfo.yml folder location to locate the SiteMerging.yml file. In the case of East, the `SiteRootFolder` is fully qualified. It is used as-is. In the case of West, the `SiteRootFolder` is not fully qualified. It is combined with the CustomerInfo.yml folder location to retrieve a fully qualified folder location for West.

To merge the East DDC to the cloud:

```
Merge-CvadAcToSite –SiteName East -CustomerInfoFileSpec \\share.
central.net\AutoConfig\CustomerInfo.yml
```

To merge the West DDC to the cloud:

```
Merge-CvadAcToSite –SiteName West -CustomerInfoFileSpec \\share.
central.net\AutoConfig\CustomerInfo.yml
```

The following is a sample SitePrefixes.yml file used in the previous example.

```
 1        East:
 2        SiteRootFolder: "\\\\share.central.net\\AutoConfig\\SiteEast"
 3        AdminScopePrefix: "East_"
 4        AdminRolePrefix: "East_"
 5        ApplicationAdminPrefix: "East_"
 6        ApplicationFolderPrefix: "" # Note that a new parent root folder
             is used instead
 7        ApplicationFolderRoot: "East"
 8        ApplicationGroupPrefix: "East_"
 9        ApplicationUserPrefix: "East_"
10        DeliveryGroupPrefix: "East_"
11        GroupPolicyPrefix: "East_"
12        HostConnectionPrefix: "East_"
13        MachineCatalogPrefix: "East_"
14        StoreFrontPrefix: "East_"
```

```
15          TagPrefix: "East_"
16          AdminScopeSuffix: "_east"
17          AdminRoleSuffix: "_east"
18          ApplicationAdminSuffix: "_east"
19          ApplicationFolderSuffix: "_east"
20          ApplicationGroupSuffix: "_east"
21          ApplicationUserSuffix: "_east"
22          DeliveryGroupSuffix: "_east"
23          GroupPolicySuffix: "_east"
24          HostConnectionSuffix: "_east"
25          MachineCatalogSuffix: "_east"
26          StoreFrontSuffix: "_east"
27          TagSuffix: "_east"
28        West:
29          SiteRootFolder: "\\\\share.central.net\\AutoConfig\\SiteWest"
30          AdminScopePrefix: "Western "
31          AdminRolePrefix: "Western "
32          ApplicationAdminPrefix: "Western "
33          ApplicationFolderPrefix: "" # Note that a new parent root folder
                 is used instead
34          ApplicationFolderRoot: "Western"
35          ApplicationGroupPrefix: "Western "
36          ApplicationUserPrefix: "Western "
37          DeliveryGroupPrefix: "Western "
38          GroupPolicyPrefix: "Western "
39          HostConnectionPrefix: "Western "
40          MachineCatalogPrefix: "Western "
41          StoreFrontPrefix: "Western "
42          TagPrefix: "Western "
43          AdminScopeSuffix: ""
44          AdminRoleSuffix: ""
45          ApplicationAdminSuffix: ""
46          ApplicationFolderSuffix: ""
47          ApplicationGroupSuffix: ""
48          ApplicationUserSuffix: ""
49          DeliveryGroupSuffix: ""
50          GroupPolicySuffix: ""
51          HostConnectionSuffix: ""
52          MachineCatalogSuffix: ""
53          StoreFrontSuffix: ""
54          TagSuffix: ""
```

If a central file share method is not used and the import, merge, or add is done from the individual DDCs, then create and replicate the SiteMerging.yml file on each DDC being migrated into the cloud. The default location is *%HOMEPATH%\Documents\Citrix\AutoConfig*. You must specify the −SiteName parameter to select the correct site prefixes.

Merging the sites

Citrix recommends performing the cloud operations in steps and to do a complete review of each result before doing the next cloud operation. For example, if merging three sites to a single cloud site:

1. Merge the initial site to the cloud using the appropriate `SiteName` value.
2. Review the results in the Full Configuration management interface.
3. If the results are incorrect, determine the issue and its cause, correct it, and then rerun the merge. If necessary, remove the cloud components and start from scratch by using `Remove-CvadAcFromSite` for the selected component and members. If the results are correct, continue.
4. If the initial merge is correct, merge the second site to the single cloud site.
5. Repeat steps 2 and 3.
6. If the second merge is correct, merge the third site to the single cloud site.
7. Repeat steps 2 and 3.
8. Review the resources from the user's perspective and verify that the view is in the desired state.

Remove a component using the site prefix

You can selectively remove single site components by using the prefix on the `-IncludeByName` parameter of the `Remove-CvadAcFromSite` cmdlet. In the following example, the West DDC delivery groups are not correct. To remove the delivery groups for just the West site:

`Remove-CvadAcFromSite –DeliveryGroups –IncludeByName "Western *"`

To remove all West components, run the following cmdlets in order.

`Remove-CvadAcFromSite –GroupPolicies –IncludeByName "Western *"`

`Remove-CvadAcFromSite –Applications –IncludeByName "Western *"`

`Remove-CvadAcFromSite – ApplicationGroups –IncludeByName "Western *"`

`Remove-CvadAcFromSite –DeliveryGroups –IncludeByName "Western *"`

`Remove-CvadAcFromSite –MachineCatalogs –IncludeByName "Western *"`

`Remove-CvadAcFromSite –HostConnections –IncludeByName "Western *"`

`Remove-CvadAcFromSite –Tags –IncludeByName "Western *"`

To remove group policies of the East components, use the suffix:

`Remove-CvadAcFromSite –GroupPolicies –IncludeByName "*_east"`

Migrating from cloud to cloud

December 5, 2023

Automated Configuration allows you to automate moving your cloud configuration to another cloud site or allowing you to restore your own cloud site.

Using Automated Configuration can solve many use cases:

- Syncing your site from test or stage to production
- Backing up and restoring your configuration
- Reaching resource limits
- Migrating from one region to another

In Full Configuration on Citrix Cloud, see the Backup and Restore node for information about Automated Configuration and how it can be used to migrate your configration from cloud to cloud.

Prerequisites for migrating your configuration

For backing up and restoring your configuration, you need:

- Citrix DaaS (formerly Citrix Virtual Apps and Desktops service) provisioned.
- An active resource location with Connector installed.
- Connectivity to sites accessing Citrix Cloud must be allowed and available. For more information, see System and Connectivity Requirements.

> **Note:**
>
> MCS cannot be backed up from the cloud using Automated Configuration.

Backing up your Citrix DaaS configuration

> **Important:**
>
> - You must have your CustomerInfo.yml file with your customer ID, client ID, and the secret key information included. For more information on how to retrieve your customer ID, client ID and secret key, see Generating the customer ID, client ID, and secret key. For information on how to add this information to the CustomerInfo.yml file, see Populating customer info file.
> - The ZoneMapping.yml file must include information that maps your resource locations in the cloud. For more information on how to map your zones, see Populating zone mapping file.

> • If you have host connections, you must input the corresponding info in the CvadAcSecurity.yml file.

1. Install Automated Configuration.

   > **Note:**
   >
   > For cloud-to-cloud migration, Automated Configuration can be installed on a machine having access to the internet that the administrator has direct access to.

2. Double-click the **Auto Config** icon. A PowerShell window appears.

3. Run the following command to do a backup.

   ```
   Backup-CvadAcToFile
   ```

After you run any cmdlet for the first time, an export folder with the .yml configuration files and logs is created. The folder is at *%HOMEPATH%\Documents\Citrix\AutoConfig*.

If you encounter any errors or exceptions, see the **Fixups** section in the log file.

Restoring your configuration to Citrix DaaS

1. Double-click the **Auto Config** icon. A PowerShell window appears.

2. Run the following command to do a restore.

   ```
   Restore-CvadAcToSite -RestoreFolder <folder path of the backup
   files>
   ```

Verify the expected state with the new current state.

After you run the cmdlet, an export folder with the .yml configuration files and logs is created. The folder is at *%HOMEPATH%\Documents\Citrix\AutoConfig*.

If you encounter any errors or exceptions, see the **Fixups** section in the log file.

The backup and restore process protects you from unintentional cloud site configuration changes or corruption. While Automated Configuration makes backups each time a change is made, this backup reflects the state of the cloud site configuration before the changes. Protecting yourself requires that you periodically back up your cloud site configuration and saving it in a safe place. If an undesirable change or corruption takes place, the backup can be used to fix the change or corruption at either a granular or full site configuration level.

Granular migration

> **Important:**
>
> For more information on component migration order, see Component migration order.

Restoring entire components

Restoring one component involves selecting one or more component parameters.

To restore the entire delivery group and machine catalog components, follow this example:

```
Restore-CvadAcToSite -RestoreFolder %HOMEPATH%\Documents\Citrix\
AutoConfig/Backup_yyyy_mm_dd_hh_mm_ss
```

Restoring component members

Restoring one or more component members makes use of the `IncludeByName` feature. The `Restore` cmdlet is invoked with the `RestoreFolder` parameter along with the selected single component and the inclusion list.

To restore two group policies from a backup, follow this example:

```
Restore-CvadAcToSite -RestoreFolder %HOMEPATH%\Documents\Citrix\
AutoConfig/Backup_yyyy_mm_dd_hh_mm_ss

-GroupPolicies -IncludeByName Policy1,Policy2

-DeliveryGroups -MachineCatalogs
```

Restoring the entire cloud site configuration

Restoring the full cloud site configuration means selecting all components to restore.

To restore the entire cloud site configuration, follow this example:

```
Restore-CvadAcToSite -RestoreFolder %HOMEPATH%\Documents\Citrix\
AutoConfig/Backup_yyyy_mm_dd_hh_mm_ss
```

Activating sites

Site activation allows you to control which site is active and controls your resources. For more information, see Activating sites.

Automated configuration tool cmdlets

December 5, 2023

This page lists all the cmdlets and parameters supported by the tool.

All cmdlets take parameters having one of the following types.

- String
- List of strings
- Boolean: `$true` or `$false`
- SwitchParameter: presence of the parameter means `$true`; absence of the parameter means `$false`

Note:

SwitchParameter is the preferred method for true or false selections but booleans are still used in the tool due to legacy issues.

The following table is a summary of all cmdlets. See each individual section to find what parameters each cmdlet supports.

Category	Cmdlet	Description
On-premises to cloud migration	`Export-CvadAcToFile`	Export on-premises files to YAML files.
		`Import-CvadAcToSite`
		`Merge-CvadAcToSite`
		`New-CvadAcToSite`
		`Sync-CvadAcToSite`
		Granular migration For components, use parameters with commands above. Examples: `MachineCatalogs`, `Tags`. For component names, use parameters with commands above. Examples: `IncludeByName`, `ExcludeByName`.

Category	Cmdlet	Description
Cloud to cloud cmdlets	`Backup-CvadAcToFile`	Backs up all the configuration from your cloud site.
	`Restore-CvadAcToSite`	
	`Remove-CvadAcFromSite`	
		Granular migration For components, use parameters with commands above. Examples: `MachineCatalogs`, `Tags`. For component names, use parameters with commands above. Examples: `IncludeByName`, `ExcludeByName`.
Other basic cmdlets	`Compare-CvadAcToSite`	Compares the on-premises .yml files with the cloud configuration.
Prerequisites-related cmdlets	`New-CvadAcCustomerInfoFile`	Create a customer info file.
	`Set-CvadAcCustomerInfoFile`	
Support and troubleshooting cmdlets	`New-CvadAcZipInfoForSupport`	Zips all log and .yml files in a single zip file to send to Citrix for support.
	`Get-CvadAcStatus`	
	`Test-CvadAcConnectionWithSite`	
	`Find-CvadAcConnector`	
	`Get-CvadAcCustomerSites`	

Category	Cmdlet	Description
	`New-CvadAcTemplateToFile`	
	`Show-CvadAcDocument`	
	`Find-CvadAcInFile`	
Site activation cmdlets	`Set-CvadAcSiteActiveStateOnPrem`	Sets the on-premises site state to either active or inactive.
	`Set-CvadAcSiteActiveStateCloud`	
Merging multiple on-premises sites cmdlets	`New-CvadAcSiteMergingInfo`	Creates a site merging prefix/suffix info set.
	`Set-CvadAcSiteMergingInfo`	
	`Remove-CvadAcSiteMergingInfo`	

For more information on parameters and how to use them, see Granular migration parameters.

Basic cmdlets

On-premises to cloud cmdlets

- `Export-CvadAcToFile` - Export on-premises files to YAML files.

  Exports configuration from your on-premises setup. This is the default export operation for Automated configuration. No modifications are made to the on-premises site configuration. Exported files are placed in the directory *%HOMEPATH%\Documents\Citrix\AutoConfig* in a uniquely named **Export** subfolder. The folder *%HOMEPATH%\Documents\Citrix\AutoConfig* always contains the latest exported on-premises site configuration.

  Parameters:

Name	Description	Required?	Type
Migrate by components	See Migrate by components		SwitchParameters
Filtering by object names	See Filtering by object names		List of strings
`TargetFolder`	Specifies the export destination folder.		String
`Locale`	Specifies the language of human-readable text that can be exported.		String
`Quiet`	Suppress logging to the console.		SwitchParameter
`AdminAddress`	Specifies the Delivery Controller's DNS or IP address when the export is not being run on the Delivery Controller.		String
`CheckUserAndMachines`	Verifies if users and machines are in Active Directory. Users and machines that are not in Active Directory might result in import failures.		`$true` or `$false`
`ZipResults`	Zips backup up YAML files into a single zip file. The file is in the same folder as the backed up YAML files and has the same name as the folder.		SwitchParameter

Returns:

- – See Cmdlet return values

There are three ways of importing data into the cloud. Running specific cmdlets can result in one of the three combinations of actions on the cloud site:

- Add, Update, and Delete
- Add and Update only
- Add only

Cmdlet	Add	Update	Delete
Import	X	X	X
Merge	X	X	
New	X		

- `Import-CvadAcToSite` - Import YAML files to the cloud. Supports create, update and delete operations.

Imports all the on-premises files to the cloud. This command ensures that the cloud end state is identical to the on-premises state. This option deletes any changes that exist in the cloud. Imported site configuration files are sourced from *%HOMEPATH%\Documents\Citrix\AutoConfig*. *Use with caution.*

Parameters:

Name	Description	Required?	Type
Migrate by components	See Migrate by components.		SwitchParameters
Filtering by object names	See Filtering by object names.		List of strings
Cloud-accessing parameters	See Cloud-accessing parameters.		SwitchParameters
SourceFolder	Identifies a substitute root folder for *%HOMEPATH%\Documents\Citrix\AutoConfig*.		String
Locale	Specifies the language of human-readable text that can be exported.		String
Quiet	Suppress logging to the console.		SwitchParameter
DisplayLog	Displays the log file at the completion of the cmdlet. Set to $false to suppress the log display.		$true or $false
Merge	When set to $true, only adds components to the cloud site. Components are not removed. Set to $false to remove components.		$true or $false
AddOnly	When set to $true, adds only new components, does not update or delete existing components. Set to $false to allow updates and deletions. Merge is ignored when this parameter is $true.		$true or $false
MergePolicies	Merge policy settings and filters. Merging occurs only when a policy being imported already exists in the cloud DDC. The result of merging policies is that the cloud DDC policies contain the settings and filters it already had in addition to any new settings and filters being imported. Note that when setting and filter collisions occur, the imported values take precedence.		SwitchParameter
OnErrorAction	See OnErrorAction parameter.		String

Returns:

- See Cmdlet return values

- `Merge-CvadAcToSite` - Import YAML files to the cloud. Supports create and update operations.

  Merges the on-premises files to the cloud, but does *not* delete any components in the cloud or the on-premises site. This preserves changes already made in the cloud. If a component exists in Citrix Cloud with the same name, this command can modify that component. This is the default import operation for Automated configuration. Merged site configuration files are sourced from *%HOMEPATH%\Documents\Citrix\AutoConfig*.

  Parameters:

Name	Description	Required?	Type
Migrate by components	See Migrate by components.		SwitchParameters
Filtering by object names	See Filtering by object names.		List of strings
Cloud-accessing parameters	See Cloud-accessing parameters.		SwitchParameters
`SourceFolder`	Identifies a substitute root folder for *%HOMEPATH%\Documents\Citrix\AutoConfig*.		String
`Locale`	Specifies the language of human-readable text that can be exported.		String
`Quiet`	Suppress logging to the console.		SwitchParameter
`DisplayLog`	Displays the log file at the completion of the cmdlet. Set to `$false` to suppress the log display.		`$true` or `$false`
`Merge`	When set to `$true`, only adds components to the cloud site. Components are not removed. Set to `$false` to remove components.		`$true` or `$false`
`AddOnly`	When set to `$true`, adds only new components, does not update or delete existing components. Set to `$false` to allow updates and deletions. `Merge` is ignored when this parameter is `$true`.		`$true` or `$false`

Name	Description	Required?	Type
MergePolicies	Merge policy settings and filters. Merging occurs only when a policy being imported already exists in the cloud DDC. The result of merging policies is that the cloud DDC policies contain the settings and filters it already had in addition to any new settings and filters being imported. Note that when setting and filter collisions occur, the imported values take precedence.		SwitchParameter
OnErrorAction	See OnErrorAction parameter.		String

Returns:

- See Cmdlet return values

- New-CvadAcToSite - Import YAML files to the cloud. Supports create and update operations.

  Imports on-premises site configuration to the cloud but only adds new components. Existing cloud site components are neither updated nor deleted. Use this command if your existing cloud site components must remain unchanged.

  Parameters:

Name	Description	Required?	Type
Migrate by components	See Migrate by components.		SwitchParameters
Filtering by object names	See Filtering by object names.		List of strings
Cloud-accessing parameters	See Cloud-accessing parameters.		SwitchParameters
SourceFolder	Identifies a substitute root folder for %HOMEPATH%\Documents\Citrix\AutoConfig.		String
Locale	Specifies the language of human-readable text that can be exported.		String
Quiet	Suppress logging to the console.		SwitchParameter
DisplayLog	Displays the log file at the completion of the cmdlet. Set to $false to suppress the log display.		$true or $false
OnErrorAction	See OnErrorAction parameter.		String

Name	Description	Required?	Type

Returns:

- – See Cmdlet return values

- `Sync-CvadAcToSite` - Export and import in one step.

  Sync performs both an export and import in one step. Use the `SourceTargetFolder` parameter to specify the export/import destination folder.

  Parameters:

Name	Description	Required?	Type
Migrate by components	See Migrate by components		SwitchParameters
Filtering by object names	See Filtering by object names		List of strings
Cloud-accessing parameters	See Cloud-accessing parameters		SwitchParameters
`SourceTargetFolder`	Specifies the export/import destination folder.		String
`Locale`	Specifies the language of human-readable text that can be exported.		String
`AdminAddress`	Specifies the delivery controller's DNS or IP address when the export is not being executed on the delivery controller.		String
`Quiet`	Suppress logging to the console.		SwitchParameter
`DisplayLog`	Displays the log file at the completion of the cmdlet. Set to `$false` to suppress the log display.		`$true` or `$false`
`Merge`	When set to `$true`, only adds components to the cloud site. Components are not removed. Set to `$false` to remove components.		`$true` or `$false`
`AddOnly`	When set to `$true`, adds only new components, does not update or delete existing components. Set to `$false` to allow updates and deletions. `Merge` is ignored when this parameter is `$true`.		`$true` or `$false`

Name	Description	Required?	Type
MergePolicies	Merge policy settings and filters. Merging occurs only when a policy being imported already exists in the cloud DDC. The result of merging policies is that the cloud DDC policies contain the settings and filters it already had in addition to any new settings and filters being imported. Note that when setting and filter collisions occur, the imported values take precedence.		SwitchParameter

Returns:

– See Cmdlet return values

Cloud to cloud cmdlets

- Backup-CvadAcToFile - Backs up all the configuration from your cloud site.

  Exports your cloud configuration to .yml files. This backup can be used in a backup and restore process to restore lost components.

  Parameters:

Name	Description	Required?	Type
Migrate by components	See Migrate by components		SwitchParameters
Cloud-accessing parameters	See Cloud-accessing parameters		SwitchParameters
TargetFolder	Specifies the export destination folder.		String
Locale	Specifies the language of human-readable text that can be exported.		String
Quiet	Suppress logging to the console.		SwitchParameter
DisplayLog	Displays the log file at the completion of the cmdlet. Set to $false to suppress the log display.		$true or $false
ZipResults	Zips backup up YAML files into a single zip file. The file is in the same folder as the backed up YAML files and has the same name as the folder.		SwitchParameter

Returns:

– See Cmdlet return values

- `Restore-CvadAcToSite` - Restores backup YAML files to the cloud site. This cloud site can be the same or different than the source cloud site.

  Restores the cloud site to the previous configuration. Imported files are sourced from the folder specified using the `-RestoreFolder` parameter, which identifies the folder containing the .yml files to restore to the cloud site. This must be a fully qualified folder specification. This cmdlet can be used for reverting to your previous configuration or for backing up and restoring your cloud site. This command can add, delete, and update your cloud site.

  Parameters:

Name	Description	Required?	Type
Migrate by components	See Migrate by components.		SwitchParameters
Filtering by object names	See Filtering by object names.		List of strings
Cloud-accessing parameters	See Cloud-accessing parameters.		SwitchParameters
`RestoreFolder`	Identifies the folder containing the .yml files to restore to the cloud site. This must be a fully qualified folder specification.		String
`Locale`	Specifies the language of human-readable text that can be exported.		String
`Quiet`	Suppress logging to the console.		SwitchParameter
`DisplayLog`	Displays the log file at the completion of the cmdlet. Set to `$false` to suppress the log display.		`$true` or `$false`
`Merge`	When set to `$true`, only adds components to the cloud site. Components are not removed. Set to `$false` to remove components.		`$true` or `$false`
`AddOnly`	When set to `$true`, adds only new components, does not update or delete existing components. Set to `$false` to allow updates and deletions. `Merge` is ignored when this parameter is `$true`.		`$true` or `$false`

Name	Description	Required?	Type
MergePolicies	Merge policy settings and filters. Merging occurs only when a policy being imported already exists in the cloud DDC. The result of merging policies is that the cloud DDC policies contain the settings and filters it already had in addition to any new settings and filters being imported. Note that when setting and filter collisions occur, the imported values take precedence.		SwitchParameter
OnErrorAction	See OnErrorAction parameter.		String

Returns:

- See Cmdlet return values

- Remove-CvadAcFromSite —Remove component members from cloud.

  Can reset the entire site or remove member items from a component (For example, removing one machine catalog from the list of catalogs). This can be used when coupled with the IncludeByName parameter to selectively remove specific members.

  Parameters:

Name	Description	Required?	Type
Migrate by components	See Migrate by components		SwitchParameters
Filtering by object names	See Filtering by object names		List of strings
Cloud-accessing parameters	See Cloud-accessing parameters		SwitchParameters
Quiet	Suppress logging to the console.		SwitchParameter
DisplayLog	Displays the log file at the completion of the cmdlet. Set to $false to suppress the log display.		$true or $false

Returns:

- See Cmdlet return values

Other basic cmdlets

- `Compare-CvadAcToSite` - Compares the on-premises .yml files with the cloud configuration, producing a report of changes that are made by an `Import`, `Merge`, or `Restore` cmdlet.

  Parameters:

Name	Description	Required?	Type
Migrate by components	See Migrate by components.		SwitchParameters
Filtering by object names	See Filtering by object names.		List of strings
Cloud-accessing parameters	See Cloud-accessing parameters.		SwitchParameters
SourceFolder	Identifies a substitute root folder for *%HOMEPATH%\Documents\Citrix\AutoConfig*.		String
Locale	Specifies the language of human-readable text that can be exported.		String
Quiet	Suppress logging to the console.		SwitchParameter
DisplayLog	Displays the log file at the completion of the cmdlet. Set to $false to suppress the log display.		$true or $false
Merge	When set to $true, only adds components to the cloud site. Components are not removed. Set to $false to remove components.		$true or $false
AddOnly	When set to $true, adds only new components, does not update or delete existing components. Set to $false to allow updates and deletions. Merge is ignored when this parameter is $true.		$true or $false
OnErrorAction	See OnErrorAction parameter.		String

  Returns:

  – See Cmdlet return values

Granular migration parameters

Migrate by components

The following components can be specified with cmdlets supporting them. The `All` option is automatically selected when no component parameters are specified. To avoid errors, we recommend that you migrate components in the following order:

- `All`
- `Tags`
- `AdminRoles`
- `AdminScopes`
- `HostConnections`
- `MachineCatalogs`
- `StoreFronts`
- `DeliveryGroups`
- `ApplicationGroups`
- `ApplicationFolders`
- `Applications`
- `GroupPolicies`
- `UserZonePreference`

Filtering by object names

Migrate by component names The `IncludeByName` and `ExcludeByName` parameters enable including and excluding component members in cmdlets by name. Only one component (for example, delivery groups) can be chosen at a time in any of the supported cmdlets. If a component member is in both areas, exclude overrides any other parameter and an entry is made in the log fixup list identifying the component and member name that was excluded.

`IncludeByName` and `ExcludeByName` take a list of component member names. Any name can contain one or more wildcards. Two types of wildcards are supported. The list of component member names must be enclosed in single-quotes when any member name contains special characters.

- * Matches any number of characters
- ? Matches a single character

`IncludeByName` and `ExcludeByName` can also take a file containing a list of members where each member can be explicit or contain wildcards. Each line in the file can contain one member. Leading and trailing spaces are trimmed from the member name. The file name must be preceded by the @ sign and be surrounded by single quotes (a PowerShell requirement so the @ is not reinterpreted). Multiple files can be listed in addition to being mixed with member names.

One example of merging all delivery groups whose names begin with `DgSite1` and contain `Home2` would be written:

```
Merge-CvadAcToSite –DeliveryGroups –IncludeByName DgSite1*,*Home2*
```

By Delivery Group Name `ByDeliveryGroupName` filters by the delivery group name for applications and application groups. This parameter is always an inclusion list identifying members to include based on their delivery group association.

`ByDeliveryGroupName` takes a list of delivery group names. Any name can contain one or more wildcards. Two types of wildcards are supported.

- * matches any number of characters
- ? matches a single character

The following example merges all applications that reference all delivery group names beginning with `EastDg`.

```
Merge-CvadAcToSite –Applications –ByDeliveryGroupName EastDg*
```

Exclude Disabled `ExcludeDisabled` filters out from import operations all applications and application groups that are disabled. `ExcludeDisabled` defaults to **false**, meaning all applications and application groups are imported regardless of their enabled state.

By Machine Name `ByMachineName` filters by the machine name for machine catalogs and delivery groups. This parameter is always an inclusion list identifying members to include based on their machine name association.

`ByMachineName` takes a list of machine names where any name can contain one or more wildcards. Two types of wildcards are supported.

- * matches any number of characters
- ? matches a single character

When exporting or importing and using `ByMachineName` and a machine name filter results in no machines in the machine catalog or delivery group, the machine catalog or delivery group is excluded from the export or import.

> **Note:**
>
> Use of `ByMachineName` in any import type cmdlet results in `MergeMachines` being set to `$true`.

Merge Machines `MergeMachines`, when set to `$true`, instructs the import operation to add machines only to the machine catalog or delivery group. Machines are not removed, allowing for incremental additive operations.

`MergeMachines` defaults to false meaning machines are removed if they are not present in the machine catalog or delivery group .yml file. `MergeMachines` is set to `$true` when `ByMachineName` is used but can be overridden by setting `MergeMachines` to false.

Prerequisites-related cmdlets

- `New-CvadAcCustomerInfoFile` - Create a customer info file. By default, the customer info file is located at *%HOMEPATH%\Documents\Citrix\AutoConfig*.

  Parameters:

Name	Description	Required?	Type
CustomerId	Customer's ID.	x	String
ClientId	Customer's client ID created on Citrix Cloud. The CustomerId and Secret must be specified when using this parameter.	Conditionally	String
Secret	Customer's secret key created on Citrix Cloud. The CustomerId and ClientId must be specified when using this parameter.	Conditionally	String
Environment	Production, ProductionGov, or ProductionJP environment.		Enumeration
LogFileName	Change the log file prefix from CitrixLog to something else.		String
AltRootUrl	Use only under the direction of Citrix.		String
StopOnError	Stops the operation upon first error.		$true or $false
TargetFolder	Use the specified folder as the root folder instead of *%HOMEPATH%\Documents\Citrix\AutoConfig*.		String
Locale	Use the specified local instead of the locale derived from the system the tool is run on.		String

Name	Description	Required?	Type
Editor	Use the specified editor to display the log at the completion of each cmdlet. Notepad.exe is the default editor. This parameter must include the fully qualified file specification to the editor and the editor must take the log file spec as its only parameter.		String
SecurityCsvFileSpec	The fully qualified file specification pointing to the SecurityClient.csv file downloaded from Citrix Identity and Access Management. The CustomerId must be specified when using this parameter.		String

Returns:

- See Cmdlet return values

- Set-CvadAcCustomerInfoFile - Update an existing customer info file. Only cmdlet specified parameters are changed. All unspecified parameter values in the CustomerInfo.yml file are unchanged.

Parameters:

Name	Description	Required?	Type
CustomerId	Customer's ID.		String
ClientId	Customer's client ID created on Citrix Cloud.		String
Secret	Customer's secret key created on Citrix Cloud.		String
Environment	Production, ProductionGov, or ProductionJP environment.		Enumeration
LogFileName	Change the log file prefix from CitrixLog to something else.		String
StopOnError	Stops the operation upon first error.		$true or $false
TargetFolder	Use the specified folder as the root folder instead of *%HOMEPATH%\Documents\Citrix\AutoConfig*.		String
Locale	Use the specified local instead of the locale derived from the system the tool is run on.		String

Name	Description	Required?	Type
Editor	Use the specified editor to display the log at the completion of each cmdlet. Notepad.exe is the default editor. This parameter must include the fully qualified file specification to the editor and the editor must take the log file spec as its only parameter.		String
SecurityCsvFileSpec	The fully qualified file specification pointing to the SecurityClient.csv file downloaded from Citrix Identity and Access Management. The CustomerId must be specified when using this parameter.		String

Returns:

- See Cmdlet return values

Prerequisites-related parameters

Along with the cloud accessing parameters, the following parameters can be used with the prerequisites-related cmdlets:

- `Environment` –Production or ProductionGov environment.
- `LogFileName` –Change the log file prefix from CitrixLog to something else.
- `StopOnError` –Stops the operation upon first error.
- `AlternateRootFolder` –Use the specified folder as the root folder instead of *%HOME-PATH%\Documents\Citrix\AutoConfig*.
- `Locale` –use the specified local instead of the locale derived from the system the tool is run on.
- `Editor` –use the specified editor to display the log at the completion of each cmdlet. Notepad.exe is the default editor. This parameter must include the fully qualified file specification to the editor and the editor must take the log file spec as its only parameter.

Support and troubleshooting cmdlets

- `New-CvadAcZipInfoForSupport` - Zips all log and .yml files in a single zip file to send to Citrix for support. Customer sensitive information (CustomerInfo.yml and CvadAcSecurity.yml) is not included in the zip. The Icon.yml file is also excluded due to its size. The

zip file is placed in *%HOMEPATH%\Documents\Citrix\AutoConfig* and named *CvadAcSupport_yyyy_mm_dd_hh_mm_ss.zip*, based on the date and timestamp. This zip file can also act as a backup.

Parameters:

Name	Description	Required?	Type
TargetFolder	Specifies a target folder to create and save the zip file.		String
Quiet	Suppress logging to the console.		SwitchParameter

Returns:

- Zip file with zip file name and location is displayed on the command prompt.

- Get-CvadAcStatus - Use to test connectivity and to ensure all prerequisites are met. Returns information about the tool like version number and connectivity with the cloud and connector status.

Parameters:

Name	Description	Required?	Type
Cloud-accessing parameters	See Cloud-accessing parameters		SwitchParameters
SiteId	Identifies the site to connect to.		String
AdminAddress	This is the DNS or IP address of the on-premises Delivery Controller used to verify the admins access level. This is required if the tool is not being run on a Delivery Controller.		String

Returns:

- Displays the results for each item.

- Test-CvadAcConnectionWithSite – Test the connection with the cloud site to verify that the communication connection is working. This cmdlet uses the cloud accessing parameters or the CustomerInfo.yml file to specify the customer connection information.

Parameters:

Name	Description	Required?	Type
Cloud-accessing parameters	See Cloud-accessing parameters		SwitchParameters
`Quiet`	Suppress logging to the console.		SwitchParameter

Returns:

- Test results are displayed on the command line.

- `Find-CvadAcConnector` - Locates existing connectors and determines their running state. This cmdlet uses information from the CustomerInfo.yml file or the customer ID parameter to locate the customer's connectors.

  Parameters:

Name	Description	Required?	Type
`CustomerInfoFileSpec`	The file specification pointing to a customer information file to override the default location and name. This parameter is ignored when the `CustomerId` parameter is provided.		String
`CustomerId`	The customer's ID. This parameter overrides the same value in the CustomerInfo.yml file.		String

Returns:

- Results are shown on the command line.

- `Get-CvadAcCustomerSites` - Returns the list of all the customer sites. This cmdlet uses the cloud accessing parameters or the CustomerInfo.yml file to specify the customer connection information.

  Parameters:

  - See Cloud-accessing parameters

  Returns:

  - Displays a list of found customer site IDs.

- `New-CvadAcTemplateToFile` – Creates a template file for selected components, allowing you to manually create an import file.

  Parameters:

Name	Description	Required?	Type
Migrate by components	See Migrate by components		SwitchParameters
`TargetFolder`	Specifies the export destination folder.		String

Returns:

- See Cmdlet return values

- `Show-CvadAcDocument` - Displays this documentation in the default browser.

  Parameters:

  - None.

  Returns:

  - Display this webpage in the default web browser.

- `Find-CvadAcInFile` - Find in file searches component YAML files looking for members matching a one or more names that may contain wildcards. The result is a report of found members. Find in file can only search one component at a time. Find in file searches all YAML files in the current folder and all subfolders. Use `FindSourceFolder` to limit the number of files to search.

  Parameters:

Name	Description	Required?	Type
Migrate by components	See Migrate by components. Note: The `-All` value is not valid.		SwitchParameters
`IncludeByName`	A list specifying the names of delivery groups to include when setting the site active state to active. The '*'and '?'wildcards are supported in names.		List of strings
`Unique`	Report only unique-found members.		SwitchParameter
`IncludeYaml`	Include the member specific YAML.		SwitchParameter
`FindSourceFolder`	The folder find begins searching in.		String

Name	Description	Required?	Type
DisplayLog	Displays the log file at the completion of the cmdlet. Set to $false to suppress the log display.		SwitchParameter
Quiet	Suppress logging to the console.		SwitchParameter

Return:

– Creates a report containing found members for the specified component.

Site activation cmdlets

For more information on activating sites and the usage of these cmdlets, see Activating sites.

- Set-CvadAcSiteActiveStateOnPrem - Sets the on-premises site state to either active or inactive.

  Parameters:

Name	Description	Required?	Type
Cloud-accessing parameters	See Cloud-accessing parameters		SwitchParameters
SiteActive	When present, sets the on-premises site to active removing the maintenance mode from all delivery groups. When this parameter is not present, maintenance mode is set on all delivery groups.		SwitchParameter
IncludeByName	A list specifying the names of delivery groups to include when setting the site active state to active. The '*'and '?'wildcards are supported in names.		List of strings
ExcludeByName	A list specifying the names of delivery groups to exclude when setting the site active state to active. The '*'and '?'wildcards are supported in names.		List of strings
Quiet	Suppress logging to the console.		SwitchParameter

Name	Description	Required?	Type
DisplayLog	Displays the log file at the completion of the cmdlet. Set to $false to suppress the log display.		$true or $false

Returns:

- See Cmdlet return values

- Set-CvadAcSiteActiveStateCloud - Sets the cloud site state to either active or inactive.

Parameters:

Name	Description	Required?	Type
Cloud-accessing parameters	See Cloud-accessing parameters		SwitchParameters
SiteActive	When present, sets the cloud site to active removing the maintenance mode from all delivery groups. When this parameter is not present, maintenance mode is set on all delivery groups.		SwitchParameter
IncludeByName	A list specifying the names of delivery groups to include when setting the site active state to active. The '*'and '?'wildcards are supported in names.		List of strings
ExcludeByName	A list specifying the names of delivery groups to exclude when setting the site active state to active. The '*'and '?'wildcards are supported in names.		List of strings
Quiet	Suppress logging to the console.		SwitchParameter
DisplayLog	Displays the log file at the completion of the cmdlet. Set to $false to suppress the log display.		$true or $false

Returns:

- See Cmdlet return values

Merging multiple on-premises sites cmdlets

For more information on site merging and usage of these cmdlets, see Merge multiple sites into a single site.

- `New-CvadAcSiteMergingInfo` - Creates a site merging prefix/suffix info set. It is not necessary to know all prefixes or suffixes at the beginning. They can be updated with `Set-CvadAcSiteMergingInfo` or by manually editing the SiteMerging.yml file.

  Parameters:

Name	Description	Required?	Type
SiteName	The name used to identify the set of prefixes/suffixes for a specific site. It can match the name of the actual site but does not need to.	x	String
Site merging parameters	See Site merging parameters		SwitchParameters
Quiet	Suppress logging to the console.		SwitchParameter

  Returns:

    – None

- `Set-CvadAcSiteMergingInfo` - Updates an existing site merging prefix/suffix info set.

  Parameters:

Name	Description	Required?	Type
SiteName	The name used to identify the set of prefixes/suffixes for a specific site. It can match the name of the actual site but does not need to.	x	String
Site merging parameters	See Site merging parameters		SwitchParameters

Name	Description	Required?	Type
Quiet	Suppress logging to the console.		SwitchParameter

Returns:

– None

- Remove-CvadAcSiteMergingInfo - Removes an existing site merging prefix/suffix info set.

  Parameters:

  – SiteName –identifies the set of site prefixes and suffixes. This is a string and is required.

  Returns:

  – None

Site merging parameters

The following parameters can be used when running the site merging cmdlets. All listed parameters are strings.

- SiteName –the name used to identify the set of prefixes/suffixes for a specific site. It can match the name of the actual site but does not need to. SiteName is a required parameter.
- AdminScopedPrefix –the prefix to apply to administrator scopes.
- ApplicationPrefix - the prefix to apply to applications.
- ApplicationFolderPrefix –the prefix to apply to application folders; ApplicationFolderPrefi can be combined with ApplicationFolderRoot.
- ApplicationFolderRoot –the new root folder to application folders. This creates an extra folder hierarchy. ApplicationFolderRoot can be combined with ApplicationFolderPrefix.
- ApplicationGroupPrefix –the prefix for application groups.
- ApplicationUserPrefix –the prefix to apply to the application name the user sees.
- ApplicationAdminPrefix –the prefix to apply to the application name the administrator sees.
- DeliveryGroupPrefix –the prefix to apply to delivery groups.
- GroupPolicyPrefix –the prefix to apply to policy names.
- HostConnectionPrefix –the prefix to apply to host connections.
- MachineCatalogPrefix –the prefix to apply to machine catalogs.
- StoreFrontPrefix –the prefix to apply to StoreFront names.

- `TagPrefix` –the prefix to apply to tags.
- `AdminScopedSuffix` –the suffix to apply to administrator scopes.
- `ApplicationSuffix` - the suffix to apply to applications.
- `ApplicationFolderSuffix` –the suffix to apply to application folders; `ApplicationFolderSuffi`
  can be combined with `ApplicationFolderRoot`.
- `ApplicationGroupSuffix` –the suffix for application groups.
- `ApplicationUserSuffix` –the suffix to apply to the application name the user sees.
- `ApplicationAdminSuffix` –the suffix to apply to the application name the administrator
  sees.
- `DeliveryGroupSuffix` –the suffix to apply to delivery groups.
- `GroupPolicySuffix` –the suffix to apply to policy names.
- `HostConnectionSuffix` –the suffix to apply to host connections.
- `MachineCatalogSuffix` –the suffix to apply to machine catalogs.
- `StoreFrontSuffix` –the suffix to apply to StoreFront names.
- `TagSuffix` –the suffix to apply to tags.
- `SiteRootFolder` –the fully qualified folder name to use for exports and imports; this can be
  a local folder or a file share.

Generic parameters

Cloud accessing parameters

All cmdlets accessing the cloud support the following extra parameters.

> **Note:**
>
> The CustomerId, ClientId, and Secret can be placed in the CustomerInfo.yml file or specified with
> the cmdlet using the following parameters. When they are specified in both places, the cmdlet
> parameters take precedence.

- `CustomerId` –The customer ID used in the Rest APIs and is required to access all Rest APIs.
  Your customer ID is found in Citrix Cloud.
- `ClientId` –The clientID created on the Citrix Cloud Identity and Access Management website.
  This is required to obtain the bearer token needed for authentication for all Rest APIs.
- `Secret` –The secret key created on the Citrix Cloud Identity and Access Management website.
  This is required to obtain the bearer token needed for authentication for all Rest APIs.
- `CustomerInfoFileSpec` –The file specification pointing to a customer information file to
  override the default location and name.

Migration mode parameters

Cmdlets modifying the cloud site configuration (`Import`, `Restore`, `Merge`, `New`, and `Sync`) support the following extra parameters to provide further flexibility.

- `CheckMode` —Performs the import operation but makes *no* changes. All expected changes are reported before the import completes. You can use this command to test your import before it occurs.
- `BackupFirst` —Backs up the cloud contents to .yml files before modifying the cloud configuration. This is enabled by default.
- `Confirm` —When true, prompts users to confirm that they want to make changes to the cloud site configuration. The `Remove` cmdlet shows a prompt due to its destructive nature. Set to false if no prompt is desired, such as running inside automated scripts. `Confirm` defaults to true.
- `SecurityFileFolder` —This is the fully qualified folder containing the CustomerInfo.yml file which might point to a local folder or a network share folder that may be under authentication control. The tool will not prompt for credentials; access to the controlled resource must be obtained before running the tool.
- `SiteName` —Specifies the site merging prefix and suffix set to use when importing.
- `SiteActive` —Specifies whether the imported site is active or inactive. By default, this parameter is set to `$false` meaning the imported site is inactive.

Log display parameters

The `Export`, `Import`, `Sync`, `Restore`, `Backup`, `Compare`, and `Remove` cmdlets display the log file when the operation completes. You can suppress the display by setting the `-DisplayLog` parameter to `$false`. Notepad.exe is used by default to display the log file. You can specify a different editor in the CustomerInfo.yml file.

`Editor: C:\Program Files\Notepad++\notepad++.exe`

Cmdlet return values

ActionResult

All cmdlets return the following value.

```
1            public class ActionResult
2            {
3
4                public bool                        Overall_Success;
5                public Dictionary<string, string>   Individual_Success;
```

```
6                    public object                        CustomResult;
7            }
```

`Overall_Success` returns a single boolean showing the overall success of the cmdlet across all selected components: true meaning successful and false meaning unsuccessful.

`Individual_Success` returns one or three values for each main component. A component's result can be Success, Failure, or Skipped. Skipped indicates that the component was not selected for execution by the cmdlet.

`CustomResult` is cmdlet specific.

CustomResult

`Import`, `Merge`, `Restore`, `Sync`, `Compare`, `Compare File`, and `Remove` return the following custom result information to a single instance of `EvaluationResultData`.

> **Note:**
>
> `Export` and `Template` cmdlets do not return a custom result.

```
1            public class EvaluationResultData
2            {
3
4                public Dictionary<string, Dictionary<string,
                     ActionResultValues >> EvaluationResults;
5                public int                  Added;
6                public int                  Updated;
7                public int                  Deleted;
8                public int                  NoChange;
9                public int                  TotalChanged;
10               public EvaluationResults    OverallResult;
11               public string               CloudBackupFolder;
12               public string               SourceBackupFolder;
13             }
14
15           Where:
16           public enum ActionResultValues
17           {
18
19               Add,
20               Update,
21               Delete,
22               Identical,
23               DoNothing
24             }
25
26           public enum EvaluationResults
27           {
28
```

```
29                          Success,
30                          Failure,
31                          Skipped
32                  }
```

`EvaluationResults` displays a list with one entry per selected component. The key is the component name and the value is a list of each component member and the action taken on that component member. Actions can be any one of the `ActionResultValues` values.

`Added`, `Updated`, `Deleted`, and `NoChange` indicate that the total number of component members added, updated, deleted, or no action taken, in that order.

`TotalChanged` is the sum of `Added`, `Updated`, and `Deleted`.

`OverallResult` is a single boolean indicating the result of the cmdlet. True indicates total success across all components and false indicates failure in processing one or more components.

`CloudBackupFolder` is the fully qualified file specification of the cloud site configuration backup before the cmdlet performing any cloud-modifying actions.

`SourceBackupFolder` is the fully qualified file specification of the source file backup made after completion of the cmdlet. By default, these files are at *%HOMEPATH%\Documents\Citrix\AutoConfig*.

PowerShell help

PowerShell help is available for each cmdlet. All parameters are documented with each cmdlet along with a brief explanation of the cmdlet. To access help for any cmdlet, type `Get-Help` in front of the cmdlet.

`Get-Help Import-CvadAcToSite`

Troubleshoot Automated configuration and additional information

December 5, 2023

> **Important:**
>
> For commonly occurring error messages for Automated configuration and corresponding solutions, see the *troubleshooting FAQ* at Knowledge Center article CTX277730.

Automated configuration tool errors

Automated configuration tool operations can sometimes produce errors. When this happens failures can occur when processing components like Machine Catalogs, Delivery Groups, or Group Policies,

for example. Using `OnErrorAction` and continuation parameters allows you to catch errors mid-processing, resolve them, and pick up where you left off.

The default `OnErrorAction` value is `StopCompEnd`. When an error occurs, the tool finishes processing the current component. No additional components are processed, and errors do not carry forward to downstream, dependent components. After you resolve any errors, you can rerun your cmdlets with any continuation parameter applied.

OnErrorAction parameter

You can define `OnErrorAction` parameter values on migration commands to control how the tool responds to errors that it finds when processing components.

This table shows parameter values and their descriptions:

Value	Description
`Continue`	Attempts to process as many of all components as possible.
`Pause`	Pauses at the end of processing and prompts you to continue or stop.
`StopCompEnd`	Attempts to process as much of the component as possible. Stops after the component is finished. (Default)
`StopImmediately`	Processing stops when an error is found.

Migration cmdlets

You can apply the `OnErrorAction` parameter to the following migration commands:

- `Compare-CvadAcToSite`
- `Import-CvadAcToSite`
- `Merge-CvadAcToSite`
- `New-CvadAcToSite`
- `Restore-CvadAcToSite`

Example: `Merge-CvadAcToSite -OnErrorAction StopImmediately`

Resume parameters

These parameters define how the tool resumes after an operation pauses or stops because of an error.

You can apply resume parameters to migration cmdlets that include one of the following `OnErrorAction` parameter values:

- `Pause`
- `StopCompEnd`
- `StopImmediately`

This table shows parameter values and their descriptions:

Value	Description
`-AllRemaining`	Requires a starting component. Processing begins at the starting component and processes all remaining components. Multiple components are processed.
`-Resume`	Uses the component from `CurrentComponent.txt` as the starting point. All remaining is set to true. Multiple components are processed.
`-Repeat`	Uses the component from `CurrentComponent.txt` as the starting point. All remaining is set to false. Only one component is processed.

The last component processed is stored in the `CurrentComponent.txt` file in the AutoConfig folder. Editing this file is not recommended.
If you specify `-Resume` or `-Repeat`, and `CurrentComponent.txt` is missing or invalid, processing stops, and you are prompted to select a component.

Setting the OnErrorAction in the CustomerInfo.yml file

You can also set `OnErrorAction` values in the `CustomerInfo.yml` file. Set the values using the following cmdlets:

- For a new file: `New-CvadAcCustomerInfoFile -OnErrorAction Continue | Pause | StopCompEnd | StopImmediately`

- For an existing file: `Set-CvadAcCustomerInfoFile -OnErrorAction Continue | Pause | StopCompEnd | StopImmediately`

Logs

Running any cmdlet results in a log file creation and an entry in the main history log file. All operation log files are placed in a backup folder. All log file names begin with `CitrixLog`, then show the auto-config operation and the date and timestamp of the cmdlet execution. Logs do not auto-delete.

The main history log is located in `*%HOMEPATH%\Documents\Citrix\AutoConfig*`, in the file named **History.Log**. Each cmdlet execution results in a main log entry containing the date, operation, result, backup, and log file locations of the execution.

You can also use the `New-CvadAcZipInfoForSupport` cmdlet to collect logs to send to Citrix for support. This cmdlet zips all log and .yml files in a single zip file. Customer sensitive information (CustomerInfo.yml and CvadAcSecurity.yml) is not included in the zip. The Icon.yml file is also excluded due to its size. The zip file is placed in *%HOMEPATH%\Documents\Citrix\AutoConfig* and named *CvadAc-Support_yyyy_mm_dd_hh_mm_ss.zip*, based on the date and timestamp. This zip file can also act as a backup.

Each log file includes the following:

- The name of the operation and whether the check mode is enabled
- The start and end date and time
- Multiple entries for each component's actions and success/failure notifications
- Summary of actions taken including various counts of created objects
- Suggested fixes where applicable
- Backup folder location where applicable
- Main log location
- Duration

Diagnostic files

Diagnostic files assist you in determining and resolving problems. The following files are created when their operation is run. They are located in the action-specific subfolder under *%HOME-PATH%\Documents\Citrix\AutoConfig*. Include these files when providing information for problem resolution support.

Export

`PoshSdk_yyyy_mm_dd_hh_mm_ss.ps1`

This file counts all Broker PowerShell SDK calls made to export the site configuration to files.

Import, Merge, Restore, Sync, Backup, Compare

`Transaction_yyyy_mm_dd_hh_mm_ss.txt`

This file documents each Rest API call and related information.

`RestApiContent_yyyy_mm_dd_hh_mm_ss.txt`

This file contains the all `Add`, `Update`, and `Delete` Rest API content.

Problems resulting from dependencies

Imports and merges might fail due to missing dependencies. Some common problems are:

1. Group Policies are missing delivery group filters. The usual causes are delivery groups that have not been imported.
2. Applications fail to import or merge. The usual cause is missing delivery groups or application groups that have not been imported.
3. Application groups are missing a RestrictToTag. The usual causes are tags that have not been imported.
4. Host connections fail. The usual cause is missing security information in the CvadAcSecurity.yml file.
5. Machine catalogs fail. The usual cause is host connections that were not imported.
6. Machines missing from machine catalogs and delivery groups. The usual cause is machines that were not found in Active Directory.
7. Users missing from delivery groups. The usual cause is users that were not found in Active Directory.

Recommendations

- Do not run more than one instance of Automated configuration at a time. Running multiple concurrent instances produces unpredictable results in the cloud site. If this occurs, rerun one instance of Automated configuration to bring the site to the expected state.
- Do not work or change data in the Manage tab in Full Configuration while running Automated configuration.
- Always visually verify the merge or import or restore results in Full Configuration to ensure that the cloud site meets expectations.

Folders

Default folder root location

All Automated configuration tool operations occur in the root folder or in subfolders inside it. The root folder is located in *%HOMEPATH%\Documents\Citrix\AutoConfig*.

Export

All exported files are placed in two folder locations, providing ease-of-use and a history of exports. Exports are always placed in the root folder. Copies are placed in a subfolder named **Export** with the date and time of the export.

The root folder always contains the most recent exported on-premises site configuration. Each **Export** subfolder contains the export done on the indicated date and time, which maintains a history of exports. You can use any **Export** subfolder to configure the cloud site. Automated configuration does not delete or modify existing export subfolders.

Import/Merge/Sync/Compare

`Import`, `Merge`, and `Compare` operations always sourced from files located in the root folder. Each operation results in the creation of a subfolder to which files in the root folder are copied, providing a history of cloud site changing source files.

Restore

The `Restore` operation uses an existing subfolder to configure the cloud site. The source folder is specified on the required `-RestoreFolder` parameter. Unlike with other commands, no new subfolder is created because the `Restore` operation uses an existing subfolder. The restore folder can be the root folder but still must be specified on the `-RestoreFolder` parameter.

Backups

Automated Configuration initializes, updates, and backs up a cloud site configuration. When used over time, many different configurations can change on the cloud site. To facilitate long-term use and preserve history changes, Automated Configuration uses a preservation scheme to save this history of changes and provide a method to restore earlier states.

Cloud site configuration backups are always made to a subfolder named **Backup** with the data and time of the backup. Automated Configuration does not delete or modify existing export subfolders.

You can use the backups to restore specific components or your entire configuration. To restore the entire delivery group and machine catalog components, use the cmdlet:

```
Restore-CvadAcToSite -RestoreFolder %HOMEPATH%\Documents\Citrix\
AutoConfig/Backup_yyyy_mm_dd_hh_mm_ss -DeliveryGroups -MachineCatalogs
```

> **Note:**
>
> The backup file information in the preceding cmdlet is based on your own backups.

To restore the entire cloud site configuration, use the cmdlet:

```
Restore-CvadAcToSite -RestoreFolder %HOMEPATH%\Documents\Citrix\
AutoConfig/Backup_yyyy_mm_dd_hh_mm_ss
```

> **Note:**
>
> The backup file information in the preceding cmdlet is based on your own backups.

Changing the default root folder

The `Export`, `Import`, `Merge`, `Sync`, and `Compare` operations can change the default root folder by using the `-AlternateFolder` parameter. The creation and management of per-operation subfolders remains the same as previously described.

Files copied to subfolders

All files having a ".yml" extension are copied to operation subfolders except for the following:

- CustomerInfo.yml
- ZoneMapping.yml
- CvadAcSecurity.yml

Automated fail-safe cloud site backups

A backup of the current cloud site configuration is made before running operations that change the configuration. This includes `Import`, `Merge`, `Sync`, and `Restore` parameters. The backup is always in a subfolder beneath the operational subfolder.

In the case of `Restore`, the backup folder is a subfolder of the folder specified on the `-RestoreFolder` parameter.

Automation

Automated configuration tool cmdlets can be run in automation scripts without administrator intervention by suppressing prompts and the display of the log results at cmdlet completion. You can also set parameters to do the same by using the CustomerInfo.yml file.

Add the following parameter to cloud modifying cmdlets to suppress the display of prompts.

```
-Confirm $false
```

Add the following parameter to cmdlets to suppress the display of log at the completion of the cmdlet.

```
-DisplayLog $false
```

Add the following parameter to cmdlets to suppress logging to the PowerShell command window.

```
-Quiet
```

As another method, the following parameters can be placed in the CustomerInfo.yml file.

```
Confirm: False
DisplayLog: False
```

Exporting from PCs other than the Delivery Controller

The Automated configuration tool uses multiple Citrix PowerShell SDKs to export the on-premises site configuration to files. These SDKs are automatically installed on the Delivery Controller, enabling the tool to run on the Delivery Controller without extra actions. When running on non-Delivery Controller machines, it is necessary to install the set of Citrix PowerShell SDKs needed by the tool. This SDK set is part of Citrix Studio which can be installed from the Citrix Virtual Apps and Desktops installation media.

> **Note:**
>
> Automated configuration cannot be run on the Cloud Connector.

Moving to Citrix Cloud Government and Japan Control Plane

The Citrix Cloud Government and Japan Control Plane environments use different access points to authenticate and allocate access tokens. This unique requirement applies to any Automated configuration tool accessing the cloud. Perform the following steps to use Automated configuration in these environments.

1. In the *%HOMEPATH%\Documents\Citrix\AutoConfig* folder, edit CustomerInfo.yml.

2. Add one of the following lines, depending on the environment you want to connect to, to CustomerInfo.yml (or change it, if already present.)

```
Environment: 'ProductionGov'
```

or

```
Environment: 'ProductionJP'
```

Automated Configuration is now able to be used on these environments.

Citrix Cloud data collection

For information on what information Citrix Cloud collects, see Citrix Cloud Services Customer Content and Log Handling.

Additional resources

Discussion forum

Visit the Citrix Discussion forum for Automated Configuration.

Video

Watch Under the Hood of the Automated Configuration Tool for Citrix Virtual Apps and Desktops on YouTube.

Training

The Cloud Learning Center contains step-by-step video guides to building a service deployment, including the tasks described in this article. See Migrating Citrix Virtual Apps and Desktops to Citrix Cloud Learning Path.