CITRIX APPLICATION DELIVERY MANAGEMENT (ADM) OVERVIEW CHEAT SHEET

ADM BASIC INORMATION & COMMON LOG FILES USED FOR TROUBLESHOOTING

CITRIX ADM OVERVIEW

Citrix ADM is a centralized management solution that provides visibility and automates management jobs that need to be executed across multiple instances. Manage and monitor Citrix products such as: Citrix ADC MPX, Citrix ADC VPX, Citrix ADC SDX, Citrix ADC CPX, Citrix Gateway, and Citrix SD-WAN.

MINIMUM SYSTEM REQUIREMENTS

RAM	32 GB
Virtual CPU	8 CPUs
Storage	120 GB (SSD Recommended)
Network	1
Throughput	1 Gbps

* VMotion or similar feature is not supported on Citrix ADM.

ADM DEPLOYMENT MODES

Type	Info
Standalone	Execute the deployment script to access the deployment configuration menu—located: **/mps/deployment_type.py**
High Availability	Active/Passive for Redundancy
Disaster Recovery Mode	When the primary site goes down, the restore script must be ran on the DR node: **/mps/scripts/pgsql/pgsql_restore_remote_backup.sh**
ADM Agent	(Separate Download image)

REQUIRED PROTOCOL/PORTS

	PORT	DETAILS
TCP	80/443	NITRO communication with Instances, GUI
	25	SMTP notifications
	22	SSH and SCP access
	5557/5558	Logstream communication (for security insight, web insight, and HDX insight)
	5454	DB synchronization in high availability mode
	27000	ADM license server port
UDP	162/161	Receive SNMP events from instances
	514	Receive syslog messages from instances
	4739	AppFlow communication from Instances
	5005	HA Heartbeats communication

COMMON ADM LOG FILES

ADM is a collection of processes running on FreeBSD
Log Files Location: /var/mps/log/

Control Subsystem **mps_control.log**	Initialize + monitor + stops other subsystems. It is responsible for restarting any subsystem if it crashes.
Service Subsystem **mps_service.log**	Inbuilt HTTP(s) request/response handler. Depending on the type of request from UI/API, it will be passed to the appropriate service subsystem
Inventory Subsystem **mps_inventory.log**	Retrieves build/system information from Instances. And also retrieves statistics from instances to show CPU/ Memory usage etc.
Config Subsystem **mps_config.log**	It processes any config change, which can be adding instances or any other operations.
Event Subsystem **mps_event.log**	It raises all internal events—all SNMP traps and syslogs comes to this process for ADM and instances.
Perf Subsystem **mps_perf.log**	Responsible for performance reporting instances.
Afdecoder Subsystem **mps_afdecoder.log**	Responsible to receive AppFlow traffic from instances and process data.
Afanalytics **mps_afanalytics.log**	Responsible for analytics reporting of instances.
Agentmsgrouter **mps_agentmsgrouter.log**	This is the server side for Web Socket channel between ADM Server and ADM Agent. Any communication (NITRO/SSH/FTP/SCP) to ADC (managed using agent) will be send to process.
Agent Bulk Data Proxy **mps_abdp.log**	Agent Bulk Data Proxy. Agent receives the syslog, SNMP and analytics data from ADC and converts to SQL statements. These SQL statements are sent to ABDP, which inserts data into the database

COMMON CLI CHECKS

Capturing an ADM tcpdump:
 # tcpdump -i 1 udp and dst <DESTIPADDRESS> and <PORT#> -w <FILENAME>

Generating a Support Bundle from CLI:
 # cd /mps/scripts
 # ./techsupport.pl

Checking the ADM Version:

- On a live appliance:
  # cd /mps/
  # cat version.conf
- In a support bundle, open/var/nslog/dmesg.bootgrep for flash:
  # more dmesg.boot | grep -i flash

COMMON ADM FAILURES

Checking Upgrade Status
Run the following commands and check the output:
–pgrep -lf installmas
–pgrep -lf maintenance
–pgrep -lf join_streaming_replication
–pgrep -lf pg_basebackup

Note: If any of these processes appear running, the upgrade is still in process and should not be interrupted—do NOT restart the ADM

Database Maintenance:
/mps/mas_recovery/mas_recovery.py
Note: Run this command only on Primary node. This script gives you the option to clean DB components. Ensure a system restart after system clean up and allow +30 min for the DB to restore.

ADM UI Not loading / nsroot Login Not Working:
Service subsystem process should be running, which is responsible for GUI/UI feature.
Check running processes with the following command:
 # ps -ax | grep <service name>
Check Service subsystem log—file location:
/var/mps/log/mps_service.log
Check for any core dump files (ADM crashing) - file location:
/var/core/
Subsystem failure can be caused by memory exhaustion.
Check for swap space errors (e.g. "out of swap space") - file location:
/var/log/messages
Check if migration subsystem (mas_migration) is running:
 # ps -ax | grep mas_migration

Allow for more time logging in if schema migration is taken place. Schema migration is happening when the "alter table" entry is indicated in the **mps_control.log** file.

In normal operation, only Control subsystem is running, run **ps –aux | grep postgres** to verify if the DB is running. It is possible that a sudden reboot has happened leaving the database inconsistent.

Check for startup errors:
/var/mps/db_pgsql/data/pg_ctl.log
Look for errors in the relevant postgresql-xxx.log file:
/var/mps/db_pgsql/data/pg_log/
Allow more time for log in, if clean up is not complete.
Run **ps –aux | grep "postgres"**. Verify if any postgres process marked as starting or recovering in the postgresql-xxx.logs.

Unable to Connect to the Database / Database Queries are Failing:
- In mps_event.log, check error messages related to SQL query or "PQexec failed"