

Software Management Operations

This chapter provides information about software management operations on the system.

- Understanding the Local File System, on page 1
- Maintaining the Local File System, on page 2
- Configuring the Boot Stack, on page 7
- Upgrading the Operating System Software, on page 11
- Performing Dynamic Software Updates, on page 18
- Managing License Keys, on page 18
- Managing Local-User Administrative Accounts, on page 22
- Resetting, Stopping, Starting or Deleting VMs in the ASR 5500VPC-DIVPC-SI Instance, on page 23

Understanding the Local File System

The Management Input/Output (MIO/UMIO/MIO2) card provides control and management for the system.

The local file system is made up of files that are stored on one or more of the following:

- **/flash** - Flash memory located on the circuit board of the MIO/UMIO/MIO2, is the default storage media for the operating system software image, CLI configuration, and crash log files used by the system.
- **/usb1** - This device is available when a USB memory stick is inserted on the front panel of the active MIO/UMIO/MIO2.
- **/hd-raid** - This is the solid state hard disk array supported by the Fabric and Storage Cards (FSCs) and accessed via the active MIO/UMIO/MIO2.

File Types Used by the Local File System

The following file types can be located in the local file system:

- **Operating System Software Image File**: This binary file type is identified by its **.bin** extension. The file is the operating system that is loaded by the system upon startup or reloading. This is an executable, read-only file that cannot be modified by end users.
- **CLI Configuration File**: This file type is identified by its **.cfg** extension. These are text files that contain CLI commands that work in conjunction with the operating system software image. These files determine services to be provided, hardware and software configurations, and other functions performed by the

system. The files are typically created by the end user. You can modify the files both on and off-line and use descriptive long filenames.

- **System File**: Only one file identified by a **.sys** extension is used by the system. The boot.sys file contains system-specific information, which describes how the system locates, and in what priority it loads, file groups (paired .bin and .cfg files) from its boot stack.
- **Abridged Crash Log**: The abridged crash log, identified by its **crashlog** filename, contains summary information about software or hardware failures that occur on the system. This file is located in the **/flash/crsh2/** directory on the device. You can view the contents of this file through the CLI, but you cannot modify the file.

Understanding the boot.sys File

The system uses the boot.sys file to store the prioritized boot stack parameters and file groups the system uses during startup. Modify this file only through system CLI commands and not through external means. Boot parameters contain information the system needs to locate the operating system image file, including:

- **bootmode**: This setting is typically configured to normal, and identifies how the system starts.
- **network interface configuration**: Use these optional boot method settings when you configure the system to obtain its operating system image from an external network server that is using one of the management LAN interfaces on the MIO/UMIO/MIO2 card.
- **boot stack information**: The boot stack is made up of prioritized file group entries that designate the operating system image file and the CLI configuration file to load.

When a system is started for the first time, the boot.sys file is configured to use the normal boot mode and load the operating system software image from the /flash directory.

There is no CLI configuration file contained on the local file system. This causes the system to automatically start its CLI-based Quick Setup Wizard upon the first successful boot. Refer to *Getting Started* for more information on using the Quick Setup Wizard.

Maintaining the Local File System

Use CLI commands to manage and maintain the devices that make up the local file system. Execute all the commands described in this section in the Exec Mode. Unless otherwise specified, you must have security administrator or administrator privileges to execute these commands.



Note You must be logged into the active CF VM to run the commands described below.

File System Management Commands

Use the commands in this section to manage and organize the local file system.

Important For complete information on the commands listed below, see the *Exec Mode Commands* chapter of the *Command Line Interface Reference.*

Synchronizing the File System

Commands are supported for mirroring the local file systems from the active MIO/UMIO/MIO2 to the standby MIO/UMIO/MIO2 in systems containing two cards. Use these commands to synchronize any or all of the local devices.

Important Crash log files are not synchronized when these commands are executed.

The following Exec mode command synchronizes the file systems between two MIO/UMIO/MIO2s:

```
[local]host_name# filesystem synchronize [ /flash | /usb1 | all ] [ checkonly
 ] [ from card_num | to card_num ] [ -noconfirm ]
```

Only filesystems on matching local devices are synchronized. For example, if the active MIO/UMIO/MIO2 contains two local devices (/flash and /usb1) and the standby MIO/UMIO/MIO2 contains only one local device (/flash), then synchronization only occurs on the matching local device (/flash).

The following command synchronizes the file systems on two MIO/UMIO/MIO2 flash devices.

```
[local]host_name# filsystem synchronize /flash
```

Creating Directories

Use the **mkdir** command to create a new directory on the specific local device. This directory can then be incorporated as part of the path name for any file located in the local file system.

```
[local]host_name# mkdir { /flash | /usb1 | /hd-raid } /dir_name
```

Use the following command to create a directory named *configs*:

```
[local]host_name# mkdir /flash/configs
```

Renaming Files and Directories

Use the **rename** command to change the name of a file from its original name to a different name. Remember to use the same file extension, if applicable, to ensure that the file type remains unchanged.

```
[local]host_name# rename { /flash | /usb1 | /hd-raid } /src_filename { /flash
| /usb1 | /hd-raid } /dst_filename [ -noconfirm ]]
```

Use the following command to rename a file named *iot_test.cfg* to *iot_accept.cfg* on the */flash* local device.

```
[local]host_name# rename /flash/iot_test.cfg /flash/iot_accept.cfg -noconfirm
```

Important Use the **rename** command only within the same local device. You cannot rename a file and place it onto another local device at the same time. To move a renamed file, you must use the **copy** command.

Copying Files

These instructions assume that you are at the root prompt for the Exec mode. To save your current configuration, enter the following command:

```
[local]host_name# copy from_url to_url [-noconfirm]
```

To copy a configuration file called *system.cfg* from a directory that was called *cfgfiles* to a directory named *configs_old*, enter the following command:

```
[local]host_name# copy /flash/cfgfiles/system.cfg
/flash/configs_old/system_2011.cfg
```

To copy a configuration file called *init_config.cfg* to the root directory of a TFTP server with a hostname of *config_server*, enter the following command:

```
[local]host_name# copy /flash/cfgfiles/init_confg.cfg
tftp://config_server/init_config.cfg
```

Deleting Files

The **delete** command removes a designated file from its specified location on the local file system.

Important This command does not support wildcard entries; each filename must be specified in its entirety.

Caution Do not delete the boot.sys file. If deleted, the system will not reboot on command and will be rendered inoperable.

```
[local]host_name# delete { /flash | /usb1 | /hd-raid }/filename [ -noconfirm
]
```

The following command deletes a file named *test.cfg* from the */flash* directory.

```
[local]host_name# delete /flash/test.cfg
```

Removing Directories

The **rmdir** command deletes a current directory on the specific local device. This directory can then be incorporated as part of the path name for any file located in the local file system.

Important The directory you want to remove (delete) must be empty before executing the **rmdir** command. If the directory is not empty, the CLI displays a "Directory not empty" message and will not execute.

```
[local]host_name# rmdir url /dir_name
```

The following command deletes an empty directory named *configs* in the */flash* directory.

```
[local]host_name# rmdir /flash/configs
```

Formatting Local Devices

The **format** command performs a low-level format of a local device. This operation formats the device to use the FAT16 formatting method, which is required for proper read/write functionality with the operating system.

Important Local devices that have been formatted using other methods such as NTFS or FAT32 may be used to store various operating system, CLI configuration, and crash log files. However, when placing a new local device into the MIO/UMIO/MIO2 for regular use, you should format the device via the system prior to use. This ensures that the proper file allocation table format is used, preventing any possible discrepancies between other formats used with other operating systems.

Caution The **filesystem format** command removes all files and information stored on the device.

To format a local device for use by the local file system, enter the following command:

```
[local]host_name# filesystem format { /flash | /usb1 | /hd-raid }
```

Applying Pre-existing CLI Configuration Files

A pre-existing CLI configuration file is any .cfg file created to provide utility functions (such as clearing all statistics during testing) or created off-line using a text editor. There may be pre-existing configuration files stored on the local file system that can be applied to a running system at any time.

Caution If a configuration file is applied to a system currently running another CLI configuration, any like contexts, services, logical interfaces, physical ports, IP address pools, or other configured items will be overwritten if the same command exists in the configuration file being applied. Take caution to ensure that you are knowledgeable of the contents of the file being applied and understand what the service ramifications are if a currently running command is overwritten. Also note that changes will not be saved automatically.

A CLI configuration file, or script containing CLI commands, can be applied to a running system by entering the following command at the Exec mode prompt:

```
[local]host_name# configure url [ verbose ]
```

url specifies the location of the CLI configuration file to be applied. It may refer to a local or a remote file.

The following command applies a pre-existing CLI configuration file named *clearcmds.cfg* in the */flash* directory.

```
[local]host_name# configure /flash/clearcmds.cfg
```

Viewing Files on the Local File System

This section describes how to view a variety of files.

Viewing the Contents of a Local Device

The contents, usage information, and file system directory structure of any local device can be viewed by entering the following command at the Exec mode prompt:

directory { /flash | /usb1 | /hd-raid }

Viewing CLI Configuration and boot.sys Files

The contents of CLI configuration and boot.sys files, contained on the local file system, can be viewed off-line (without loading them into the OS) by entering the following command at the Exec mode prompt:

`[local]host_name#` **show file** *url* **{ /flash | /usb1 | /hd-raid }** *filename*

Where: *url* is the path name for the location of the file and *filename* is the name of the file, including any extension.



Important Operator and inspector-level users can execute the **show file** command but cannot execute the **directory** command.

Validating an Operating System File

The operating system software image file, identified by its .bin extension, is a non-readable, non-editable file that executes on the system, creating its runtime operating system (OS).

It is important to verify a new operating system image file before attempting to load it. To accomplish this, a proprietary checksum algorithm is used to create checksum values for each portion of the application stored within the .bin file during program compilation.

This information can be used to validate the actual file against the checksum values stored within the file during its compilation. If any portion of the image file has become corrupted (for example, the file was truncated or was transferred using ASCII mode instead of binary mode), then this information is reported and the file is deemed unusable.

To validate an operating system software image file, enter the following command at the Exec mode prompt:

`[local]host_name#` **show version { /flash | /usb1 | /hd-raid } /***[directory]***/***filename* **[all]**

The output of this command displays the following information:

- Version number
- Description
- Date
- Boot Image
- Size
- Flags
- Platform

If an invalid file is found, the system displays a failure message similar to these:

```
Failure: Image /flash/image_version.bin CRC check failed!
Failure: /flash/image_version.bin, has a bad magic number
```

Configuring the Boot Stack

The boot stack consists of a prioritized listing of operating system software image-to-CLI configuration file associations. These associations determine the software image and configuration file that gets loaded during system startup or upon a reload/reboot. Though multiple associations can be configured, the system uses the association with the highest priority. If there is an error processing this association (for example, unable to locate one of the files), the system attempts to use the association with the next highest priority.

For VPC-SI and VPC-DI platforms, when the configuration file in the highest configured boot priority is not available (but the image file is), the system boots up with the configuration setup wizard after reloading instead of using the next available boot system priority. Priorities range from 1 to 100, with 1 being the highest priority. The maximum number of boot stack entries that may be configured in the boot.sys file is 10.

Boot stack information is contained in the boot.sys file, described in Understanding the boot.sys File, on page 2. In addition to boot stack entries, the boot.sys file contains any configuration commands required to define the system boot method as explained in the section that follows.

System Boot Methods

The local-boot method uses software image and configuration files stored locally on the system. On system startup or reboot, the system looks on one of its local devices or **/hd-raid** for the specific software image and accompanying configuration text file. When using the local-booting method, you only need to configure boot stack parameters.

The system can also be configured to obtain its software image from a specific external network server while it is paired with a configuration text file that resides on the system. When using network booting, you need to configure the following:

- Boot stack parameters, which define the files to use and in what priority to use them
- Boot interface and network parameters defining the remote management LAN interface and the methods to use to reach the external network server
- Network booting delay time and optional name server parameters defining the delay period (in seconds) to allow for network communications to be established, and the IP address of any Domain Name Service (DNS) name server that may be used

Detailed information on how to configure the system to use the network booting method appears in Network Booting Configuration Requirements, on page 9

Viewing the Current Boot Stack

To view the boot stack entries contained in the boot.sys file run the Exec mode **show boot** command.



Important Operator and inspector-level users can execute the **show boot** command.

The examples below shows the command output for a local booting configuration. Notice that in these examples both the image file (operating system software) and configuration file (CLI commands) are located on the **/flash** device.

Important The StarOS image filename format "asr5500-*image_number*.bin".

Example :

```
boot system priority 18 \
    image /flash/16-1-builds/asr5500-16.1.3.bin \
    config /flash/general_config.cfg

boot system priority 19 \
    image /flash/16-1-builds/asr5500-16.1.1.bin \
    config /flash/general_config_3819.cfg

boot system priority 20 \
    image /flash/16-1-builds/asr5500-16.1.0.bin \
    config /flash/general_config_3665.cfg
```

The example below shows the output for a combination network booting and local booting configuration. Notice in this example that the first two boot stack entries (Priorities 18 and 19) load the image file (operating system software) from an external network server using the Trivial File Transfer Protocol (TFTP), while all configuration files are located on the **/flash** device.

Also notice the boot network interface and boot network configuration commands located at the top of the boot stack. These commands define what remote management LAN interface(s) to use and information about communicating with the external network server that hosts the operating system software image file.

```
boot networkconfig static ip address mio1 192.168.1.150 netmask 255.255.255.0
boot delay 15
boot system priority 18 image tftp://192.168.1.161/tftpboot/image_version.bin \config
/flash/general_config.cfg
boot system priority 19 image tftp://192.168.1.161/tftpboot/image_version.bin \config
/flash/general_config.cfg
boot system priority 20 image /flash/image_version.bin \config /flash/general_config.cfg
```

To identify the boot image priority that was loaded at the initial boot time enter:

show boot initial-config

The example below displays the output:

```
[local]host_name# show boot initial-config
Initial (boot time) configuration:
      image tftp://192.168.1.161/tftpboot/image_version.bin \
       config /flash/config_name.cfg
      priority 1
```

Adding a New Boot Stack Entry

Important Before performing this procedure, verify that there are less than 10 entries in the boot.sys file and that a higher priority entry is available (i.e. that minimally there is no priority 1 entry in the boot stack). Refer to *Viewing the Current Boot Stack* for more information.

If priority 1 is in use, then you must renumber the existing entry(ies) to ensure that at least that priority is available. The maximum number of boot stack entries that can be contained in the *boot.sys* file is 10. If there are already 10 entries in the boot stack, you must delete at least one of these entries (typically, the lowest priority) and, if necessary, renumber some or all of the other entries before proceeding. Refer to Deleting a Boot Stack Entry, on page 9 for more information.

This procedure details how to add new boot stack entries to the *boot.sys* file. Make sure you are at the Exec mode prompt and enter the following commands:

```
configure
 boot system priority number image image_url config cfg_url
```

The following command creates a new boot stack entry, using a boot priority of *3*.

```
boot system priority 3 image /flash/image_filename.bin config
/flash/config_name.cfg
```



Important Boot stack changes saved to the boot.sys file are not executed until the system is rebooted.

Synchronize the local file systems on the MIO/UMIO/MIO2s with the following command:

```
filesystem synchronize all
```

Deleting a Boot Stack Entry

This procedure details how to remove an individual boot stack entry from the boot.sys file. Make sure you are at the Exec mode prompt and enter the following commands:

```
configure
  no boot system priority number
```

Where *number* specifies the boot priority used for the boot stack entry. This command removes that specific entry from the boot stack, causing the boot.sys file to be overwritten.

Network Booting Configuration Requirements

Configuring the Boot Interface

Boot interface parameters define the MIO/UMIO/MIO2 management LAN interface that the system will use to communicate with the management network when using the network booting method.

This procedure details how to configure the boot interface for reliable communications with your network server. Make sure you are at the Exec mode prompt.

Procedure

Step 1 Enter the Global Configuration mode by entering the following command:

```
[local]host_name# configure
```

The following prompt appears:

```
[local]host_name(config)#
```

Step 2 Enter the following command:

```
[local]host_name(config)#boot interface { local-eth1 | local-eth2 } medium { auto |
speed { 10 | 100 | 1000 } duplex { full | half } } media { rj45 | sfp }
```

For complete information about the above command, see the *Global Configuration Mode Commands* chapter in the *Command Line Interface Reference*.

Use port 1 for network booting.

If the speed is manually configured, you must also configure the duplex mode. In addition, you must ensure that the network server configuration supports the speed and duplex configuration.

Network speed for MIO/UMIO/MIO2 is fixed at **1000**.

Ethernet networking rules dictate that if a device's interface is configured for auto-negotiation is communicating with a device that is manually configured to support full duplex, the first device will negotiate to the manually configured speed of the second device, but will only communicate in half duplex mode.

The media for MIO/UMIO/MIO2 port 1 is fixed at **rj45**.

Step 3 Save the configuration as described in the *Verifying and Saving Your Configuration* chapter.

Configuring the Boot Network

Boot network parameters define the protocols and IP address information for MIO/UMIO/MIO2 interfaces used to reach the external network server that hosts the operating system software image file. To configure boot network parameters, make sure you are at the Exec mode prompt.

Procedure

Step 1 Enter the Global Configuration mode by entering the following command:

```
[local]host_name(config)#configure
```

The following prompt appears:

```
[local]host_name(config)#
```

Step 2 Enter the following command:

```
[local]host_name(config)# boot networkconfig { dhcp | { { dhcp-static-fallback | static
 } ip address mio5 ip_address5 [ mio6 ip_address6 ] netmask subnet_mask [ gateway
gw_ip_address ] } }
```

For complete information about the above command, see the *Global Configuration Mode Commands* chapter in the *Command Line Interface Reference*.

The following command configures the boot network to communicate using DHCP, with a static-fallback IP address for MIO/UMIO/MIO2 in slot 5 of 209.165.200.237 and a Class C netmask.

```
[local]host_name(config)# boot networkconfig dhcp-static-fallback ip address mio5
209.165.200.237 netmask 255.255.255.224
```

The next example uses static IP addresses for MIO/UMIO/MIO2 in slot 5, which can access the external network server through a gateway whose IP address is 209.165.200.238.

```
[local]host_name(config)# boot networkconfig static ip address mio5 209.165.200.237
netmask 255.255.255.224 gateway 209.165.200.238
```

Step 3 Save the configuration as described in the *Verifying and Saving Your Configuration* chapter.

Configuring Boot Network Delay Time

An optional delay period, in seconds, can be configured for systems booting from a network. The purpose of this parameter is to allow time for external devices, such as switches, that use the Spanning Tree Protocol (STP) to determine the network route to a specified IP address.

To configure a boot network delay, enter the following command from the Global Configuration mode prompt.

```
[local]host_name(config)# boot delay time
```

Where *time* is an integer from 1 to 300 seconds before attempting to contact the external network server. If your network uses STP, a typical delay time of 30 seconds should suffice.

Important Save your configuration as described in the *Verifying and Saving Your Configuration* chapter.

Configuring a Boot Nameserver

To configure a boot nameserver address, enter the following command from the Global Configuration mode prompt.

```
[local]host_name(config)# boot nameserver ip_address
```

Where *ip_address* is the IP address of the DNS server entered in IPv4 dotted-decimal notation.

Important Save the configuration as described in the *Verifying and Saving Your Configuration* chapter.

Upgrading the Operating System Software

This section describes how to manually perform the StarOS binary image upgrade procedure.

Caution Undeploying/redeploying VPC is not supported after the bin upgrade. Deactivating VPC removes the upgraded StarOS bin image.

To upgrade the StarOS software manually:

1. *Obtain VIP Addresses for AutoVNF, CF, ESC and UEM*
2. Identify OS Release Version and Build Number, on page 12

3. Download the Software Image from the Support Site, on page 13
4. *Verify Zookeeper Database*
5. *Verify ESC Database*
6. Verify Free Space on the /flash Device, on page 13
7. Transfer StarOS Image to /flash, on page 13
8. Save the Running Configuration, on page 14
9. Synchronize File Systems, on page 15
10. Reboot the System, on page 17

Identify OS Release Version and Build Number

The operating system can be configured to provide services and perform pre-defined functions through commands issued from the CLI.

The operating system software is delivered as a single binary file (**.bin** file extension) and is loaded as a single instance for the entire system.

A starfile image must be signed with an REL key before being released. A deployable image will be signed with an REL key having a ".bin.SPA" extension, where "A" identifies the revision level of the signing key. For example, **asr5500-20.0.0.bin.SPA**. If a signing key becomes compromised, a new key is created and the revision level increments to "B".

Trusted images have been introduced. The difference between a Trusted build and a Normal build is the absence of unsecure programs ftpd, telnet and tcpdump, as well as the addition of a staros.conf file for security options. Trusted images are identifiable by the presence of "_T" in the platform name. For example, **asr5500_T-20.0.0.bin.SPA**.

To identify the StarOS software version and build information:

1. Log on to the VNF to be upgraded.
2. Enter the following Exec mode command in the StarOS command line interface:

   show version

   Example output:

```
Active Software:

  Image Version:             21.9.0.69918

  Image Build Number:        69918

  Image Description:         Deployment_Build

  Image Date:                Sun Jul 22 12:08:55 EDT 2018

  Boot Image:                /flash/staros.bin

  Source Commit ID:          94797337b6c1691541ea0dd86f2f29b0f2c3630c
```

3. Execute the following Exec mode command to display additional information about the StarOS build release.

show build

Download the Software Image from the Support Site

Access to the Cisco support site and download facility is username and password controlled. You must have an active customer account to access the site and download the StarOS image.

Verify Free Space on the /flash Device

Verify that there is enough free space on the **/flash** device to accommodate the new StarOS image file.

To verify the available space on *flash* directory:

1. Log on to the CF VM using the previously fetched VIP address.

   ssh ubuntu@<*vip-addr*>

   Command example:

   ssh ubuntu@192.168.100.38

2. Enter the following Exec mode command:

   [local]*host_name*# **directory /flash**

   The following is an example of the type of directory information displayed:

```
-rwxrwxr-x    1  root     root         7334 May    5 17:29 asr-config.cfg
-rwxrwxr-x    1  root     root         399 Jun    7 18:32 system.cfg
-rwxrwxr-x    1  root     root         10667 May 14 16:24 testconfig.cfg
-rwxrwxr-x    1  root     root         10667 Jun    1 11:21 testconfig_4.cfg
-rwxrwxr-x    1  root     root         5926 Apr    7 16:27  tworpcontext.cfg
-rwxrwxr-x    1  root     root         15534 Aug    4 13:31 test_vlan.cfg
-rwxrwxr-x    1  root     root         2482 Nov 18 11:09 gateway2.cfg
-rwxrwxr-x    1  root     root         159106048 Dec 31  2011 image_filename
1136352  /flash
Filesystem           1k-blocks      Used Available Use% Mounted on
/var/run/storage/flash/part1 3115468   1136352  30018336   4%
/mnt/user/.auto/onboard/flash
```

   Note the "Available" blocks in the last line of the display. After displaying the directory information, the CLI returns to root and the following prompt appears:

   [local]*host_name*#

Transfer StarOS Image to /flash

For StarOS-based VNF, transfer the new operating system image file to the */flash* directory on the MIO/UMIO/MIO2 using one of the following methods:

- Copy the file from a network location or local device plugged in into the MIO/UMIO/MIO2 by entering the following command:

  [local]*host_name*# **copy** *from_url to_url* **[-noconfirm]**

- Transfer the file to the */flash* device using an FTP client with access to the system.

Important Whenever transferring a operating system software image file using the file transfer protocol (FTP), the FTP client must be configured to transfer the file using binary mode. Failure to use binary transfer mode will make the transferred operating system image file unusable. FTP is not supported.

- Transfer the file to the */flash* device using an SFTP client with access to the system.

Saving a Copy of the Current Configuration File

Prior to upgrading to a new software release, you should copy and rename the current configuration file to the **/flash** device and to an off-chassis location (external memory device or network URL). This renamed copy assures that you will have a fallback, loadable configuration file should a problem be encountered during the upgrade.

Save the Running Configuration

Save the currently running, upgraded configuration prior to rebooting the chassis.

To save the boot configuration:

1. Log on to the VNF using the previously fetched VIP address.

   ssh ubuntu@*<vip-addr>*

   Command example:

   ssh ubuntu@192.168.100.38

2. *Optional*. Execute the following command in the Exec mode.

   chassis key value 1234

   ```
   Save config before reload chassis, EVEN IF the same old key value is used.
   Old config scripts will become invalid after reload.
   ```

Important This step is optional, and needed only if the chassis key is not set.

3. Save the boot configuration in the flash directory.

   save configuration /flash/system.cfg

   ```
   Warning: About to overwrite boot configuration file
   Are you sure? [Yes|No]: yes
   ```

   This will update the boot configuration to use the new bin image.

   Use the following command to check the boot configuration.

   # show boot

   ```
   Monday May 21 20:39:57 UTC 2018

   boot system priority 8 \
       image /flash/sftp/production.YYYYY.qvpc-di.bin \
   ```

```
    config /flash/sftp/tb5_vnf1_dayN.cfg

boot system priority 9 \
    image /flash/staros.bin \
    config /flash/sftp/tb5_vnf1_dayN.cfg

boot system priority 10 \
    image /flash/staros.bin \
    config /flash/system.cfg
```

4. Enter the configuration mode to change the boot priority of new StarOS bin file.

   #config

   #boot system priority 1 image /flash/updated.bin config /flash/system.cfg

   #end

5. Verify the new boot priority.

   #show boot

```
boot system priority 1 \

    image /flash/updated.bin \

    config /flash/system.cfg

boot system priority 10 \

    image /flash/staros.bin \

    config /flash/system.cfg
```

6. Verify whether the flash directory contains the boot configuration and new bin.

   dir /flash

```
total 320376
-rw-rw-r-- 1 root root 134 May 3 10:11 boot.sys
-rw-rw-r-- 1 root root 3920672 May 11 19:49 crashlog2
drwxrwxr-x 2 root root 4096 May 11 19:49 crsh2
-rw-rw-r-- 1 root root 156 May 11 19:49 module.sys
drwxrwxr-x 3 root root 4096 May 11 19:49 patch
drwxrwxr-x 2 root root 4096 May 11 19:49 persistdump
-rw-rw-r-- 1 root root 79 May 11 19:49 restart_file_cntr.txt
drwxrwxr-x 3 root root 4096 May 11 20:07 sftp
-rw-rw-r-- 1 root root 160871936 May 3 10:11 staros.bin
-rw-rw-r-- 1 root root 5199 May 11 19:57 system.cfg
-rw-rw-r-- 1 root root 163227136 May 11 20:07 updated.bin
320476 /flash
Filesystem 1K-blocks Used Available Use% Mounted on
/var/run/storage/boot1/part2
4112620 320476 3792144 8% /mnt/user/.auto/onboard/flash
```

Synchronize File Systems

To synchronize the file systems:

1. Log on to the VNF using the previously fetched VIP address.

2. Synchronize the local file systems on the management cards by entering the following command:

   `[local]host_name# filesystem synchronize all`

   Example output:

```
Updating /flash/system.cfg
********************************************************************************
Updating /flash/updated.bin
********************************************************************************
Updating /flash/sftp/yang/cisco-staros-bulkstats-config.yang
********************************************************************************
Updating /flash/sftp/yang/cisco-staros-bulkstats-schema-types.yang
********************************************************************************
Updating /flash/sftp/yang/cisco-staros-bulkstats.yang
********************************************************************************
Updating /flash/sftp/yang/cisco-staros-cli-config.yang
********************************************************************************
Updating /flash/sftp/yang/cisco-staros-confd-config.yang
********************************************************************************
Updating /flash/sftp/yang/cisco-staros-config.yang
********************************************************************************
Updating /flash/sftp/yang/cisco-staros-exec.yang
********************************************************************************
Updating /flash/sftp/yang/cisco-staros-kpi.yang
********************************************************************************
Updating /flash/sftp/yang/cisco-staros-notif.yang
********************************************************************************
Updating /flash/boot.sys
********************************************************************************
12 updated on card 2
           /flash/system.cfg
           /flash/updated.bin
           /flash/sftp/yang/cisco-staros-bulkstats-config.yang
           /flash/sftp/yang/cisco-staros-bulkstats-schema-types.yang
           /flash/sftp/yang/cisco-staros-bulkstats.yang
```

```
/flash/sftp/yang/cisco-staros-cli-config.yang
/flash/sftp/yang/cisco-staros-confd-config.yang
/flash/sftp/yang/cisco-staros-config.yang
/flash/sftp/yang/cisco-staros-exec.yang
/flash/sftp/yang/cisco-staros-kpi.yang
/flash/sftp/yang/cisco-staros-notif.yang
/flash/boot.sys
```

Reboot the System

To reboot the system (VNF):

1. Log on to the VNF using the previously fetched VIP address.

   ssh ubuntu@*<vip-addr>*

   Command example:

   ssh ubuntu@192.168.100.38

2. Enter the following Exec mode command:

   `[local]`*host_name*`#` **reload [-noconfirm]**

   As the system reboots, it loads the new operating system software image and its corresponding CLI configuration file using the new boot stack entry configured earlier.

3. *Optional for PDSN:* If you are using the IP Pool Sharing Protocol during your upgrade, refer to *Configuring IPSP Before the Software Upgrade* in the *PDSN Administration Guide*.

4. Run the following Exec mode command to display additional information about the running StarOS build release.

   show build

Restoring the Previous Software Image

If for some reason you need to undo the upgrade, perform the upgrade again except:

- Specify the locations of the upgrade software image and configuration files.

then

- Specify the locations of the original software image and configuration files.

Upgrading ICSR Chassis

The procedure for upgrading primary and backup ICSR chassis is described in *Interchassis Session Recovery*. Essentially the procedure requires upgrading the primary and standby chassis using the off-line method while each is in standby mode.

Performing Dynamic Software Updates

StarOS allows the runtime loading of plugins. All StarOS builds include a "default" baseline plugin.

This feature is currently used to dynamically update the detection logic used to filter P2P applications via the Application Detection and Control (ADC) feature.

Patching is the process used to install a plugin as an incremental update to a StarOS release. One plugin can be provided to multiple, compatible, concurrent product releases. A plugin is distributed in the form of a compressed distribution kit via the internet or by other means (USB stick, CD, etc.).

A plugin is a functional software entity that provides incremental updates to a pre-existing StarOS software component. Plugins have the characteristic of being dynamically loadable at runtime and do not require a system restart. A plugin has a name and one or more versions. All plugin names are known to the system at product release.

For complete information on the Dynamic Software Update process, refer to the *ADC Administration Guide.*

Managing License Keys

License keys define capacity limits (number of allowed subscriber sessions) and available features on your system. Adding new license keys allows you to increase capacity and add new features as your subscriber base grows.

New System License Keys

New systems are delivered with no license keys installed. In most cases, you receive the license key in electronic format (usually through e-mail).

When a system boots with no license key installed a default set of restricted session use and feature licenses is installed. The following Exec Mode command lists the license information:

```
[local]host_name# show license information
```



Important With no license key installed, the session use licenses for PDSN, HA, GGSN, and L2TP LNS are limited to 10,000 sessions.

The license keys on the ASR 5500 are stored in EEPROM on the chassis midplane. Both MIO/UMIO/MIO2s access this EEPROM when booting.

Session Use and Feature Use Licenses

Session use and feature use licenses are software mechanisms that provide session limit controls and enable special features within the system. These electronic licenses are stored in the system's configuration file that is loaded as part of the system software each time the system is powered on or restarted.

- Session use licenses limit the number of concurrent sessions that a system is capable of supporting per service type and are acquired on an as-needed basis. This allows carriers to pay only for what they are using and easily increase capacity as their subscriber base grows.
- Feature use licenses enable specific features/functionality within the system and are distributed based on the total number of sessions supported by the system.

Installing New License Keys

Use the instructions below to install a new license key.

Cutting and Pasting the Key

If you have a copy of the license, use the following configuration to cut and paste just the license key part:

Procedure

Step 1 From the Exec mode, enter the following:

```
configure
   license key license
   exit
```

license is the license key string. The license can be an alphanumeric string of 1 through 1023 characters that is case sensitive. Copy the license key as shown in the example below, including the "\ (double-quote slash). Please note: this is not a functional license.

```
"\
VER=1|C1M=000-0000-00|C1S=03290231803|C2M=11-1111-11-1|C2S=\
STCB21M82003R80411A4|DOI=0000000000|DOE=00000000|ISS=1|NUM=13459|0000000000000|
LSP=000000|LSH=000000|LSG=500000|LSL=500000\|FIS=Y|FR4=Y|FPP=Y|FCS=Y|FTC=Y|FMG=Y|
FCR=Y|FSR=Y|FPM=Y|FID=Y|SIG=MCwCF\Esnq6Bs/
XdmyfLe7rHcD4sVP2bzAhQ3IeHDoyyd6388jHsHD99sg36SG267gshssja77
end
```

Step 2 Verify that the license key just entered was accepted by entering the following command at the Exec mode prompt:

```
[local]host_name# show license key
```

The new license key should be displayed. If it is not, return to the Global configuration mode and re-enter the key using the **license key** command.

Important
An invalid license will not be accepted. A Failure error will appear in the output of the **license key** command when you attempt to configure an invalid license key. If you use the **-force** option to install an invalid license key, the license will be placed into a 30-day grace period. StarOS will generate daily syslog error messages and SNMP traps during the grace period. The output of the **show license information** command will indicate "License State" as "Not Valid".

Step 3 Verify that the license key enabled the correct functionality by entering the following command:

```
[local]host_name# show license information
```

All license keys and the new session capacity or functionality enabled should be listed. If the functionality or session capacity enabled by the new key is incorrect, please contact your service representative.

Step 4 Save your configuration as described in the *Verifying and Saving Your Configuration* chapter.

Caution
Failure to save the new license key configuration in the current CLI configuration file will result in the loss of any of the new features enabled by the license key once the system is reloaded.

Adding License Keys to Configuration Files

License keys can be added to a new or existing configuration file.



Important License key information is maintained as part of the CLI configuration. Each time a key is installed or updated, you must re-save the configuration file.

Procedure

Step 1 Open the configuration file to which the new license key commands are to be copied.

Step 2 Copy the license as shown in the example, including the "\ (double-quote slash). Please note: this is not a functional license.

```
"\
VER=1|C1M=000-0000-00|C1S=03290231803|C2M=11-1111-11-1|C2S=\STCB21M82003R80411A4|
DOI=0000000000|DOE=00000000|ISS=1|NUM=13459|00000000000000|LSP=000000|LSH=000000|
LSG=500000|LSL=500000\|FIS=Y|FR4=Y|FPP=Y|FCS=Y|FTC=Y|FMG=Y|FCR=Y|FSR=Y|FPM=Y|FID=Y|
SIG=MCwCF\Esnq6Bs/XdmyfLe7rHcD4sVP2bzAhQ3IeHDoyyd6388jHsHD99sg36SG267gshssja77
end
```

Step 3 Paste the license key into the configuration

Important
Paste the license key information at the beginning of the configuration file to ensure the system has the expected capacity and features before it configures contexts.

Step 4 Save your configuration as described in the *Verifying and Saving Your Configuration* chapter.

License Expiration Behavior

When a license expires, there is a built-in grace period of 30 days that allows normal use of the licensed session use and feature use licenses. This allows you to obtain a new license without any interruption of service.

The following Exec mode command lists the license information including the date the grace period is set to expire:

show license information

Requesting License Keys

License keys for the system can be obtained through your Cisco account representative. Specific information is required before a license key may be generated:

- Sales Order or Purchase Order information
- Desired session capacity
- Desired functionality
- Midplane (chassis) serial number

To obtain the ASR 5500 chassis serial number, at the Exec mode prompt enter the **show card hardware 5** command. Look under the "MEC" heading for the "UDI Serial Number" as shown in the example below:

```
MEC:
  Description         : MEC
  Cisco Part Number   : 73-14501-01 A0
  UDI Serial Number   : FLM154300D8
  UDI Product ID      : ASR55-MEC
  UDI Version ID      : V01
```

The ICSR license key for Active and Standby chassis are uniquely coded to each chassis. Two separate license keys are required.

Viewing License Information

To see the license detail, enter the following command from the Exec mode:

`[local]`*host_name*`#` **show license information [full | key [full]]**

Activating New License Keys

To activate new license keys, you must reboot the active CF by exciting the Exec mode **reload** command.

Note All SFs are rebooted when the **reload** command is executed on the active CF.

Deleting a License Key

Use the procedure below to delete the session and feature use license key from a configuration. You must be a security administrator or administrator.

```
configure
  no license key
  exit
show license key
```

The output of this command should display: "No license key installed".

Management Card Replacement and License Keys

License keys are stored on a midplane EEPROM in the ASR 5500 chassis. The MIO/UMIO/MIO2s share these license keys. There is no need to swap memory cards into replacement MIO/UMIO/MIO2s.

Managing Local-User Administrative Accounts

Unlike context-level administrative accounts which are configured via a configuration file, information for local-user administrative accounts is maintained in a separate file in flash memory and managed through the software's Shared Configuration Task (SCT). Because local-user accounts were designed to be compliant with ANSI T1.276-2003, the system provides a number of mechanisms for managing these types of administrative user accounts.

For additional information, see Disable AAA-based Authentication for Console and Limit local-user Login on Console/vty Lines.

Configuring Local-User Password Properties

Local-user account password properties are configured globally and apply to all local-user accounts. The system supports the configuration of the following password properties:

- **Complexity:** Password complexity can be forced to be compliant with ANSI T1.276-2003.
- **History length:** How many previous password versions should be tracked by the system.
- **Maximum age:** How long a user can use the same password.
- **Minimum number of characters to change:** How many characters must be changed in the password during a reset.
- **Minimum change interval:** How often a user can change their password.
- **Minimum length:** The minimum number of characters a valid password must contain.
- **Expiry warning:** Password expiry warning interval in days.
- **Auto-generate:** Automatically generates password with option to specify length of password.

Refer to the **local-user password** command in the *Global Configuration Mode Commands* chapter of the *Command Line Interface Reference* for details on each of the above parameters.

Configuring Local-User Account Management Properties

Local-user account management includes configuring account lockouts and user suspensions.

Local-User Account Lockouts

Local-user accounts can be administratively locked for the following reasons:

- **Login failures:** The configured maximum login failure threshold has been reached. Refer to the **local-user max-failed-logins** command in the *Global Configuration Mode Commands* chapter of the *Command Line Interface Reference* for details
- **Password Aging:** The configured maximum password age has been reached. Refer to the **local-user password** command in the *Global Configuration Mode Commands* chapter of the *Command Line Interface Reference* for details.

Accounts that are locked out are inaccessible to the user until either the configured lockout time is reached (refer to the **local-user lockout-time** command in the *Global Configuration Mode Commands* chapter of the *Command Line Interface Reference*) or a security administrator clears the lockout (refer to the **clear local-user** command in the *Exec Mode Commands* chapter of the *Command Line Interface Reference*).



Important Local-user administrative user accounts could be configured to enforce or reject lockouts. Refer to the **local-user username** command in the *Global Configuration Mode Commands* chapter of the *Command Line Interface Reference* for details.

Local-User Account Suspensions

Local-user accounts can be suspended as follows:

```
configure
  suspend local-user name
```

A suspension can be removed by entering:

```
configure
  no suspend local-user name
```

Changing Local-User Passwords

Local-user administrative users can change their passwords using the **password change** command in the Exec mode. Users are prompted to enter their current and new passwords.

Security administrators can reset passwords for local-users by entering the following command from the root prompt in the Exec mode:

```
[local]host_name# password change username name
```

name is the name of the local-user account for which the password is to be changed. When a security administrator resets a local-user's password, the system prompts the user to change their password the next time they login.

All new passwords must adhere to the password properties configured for the system.

Resetting, Stopping, Starting or Deleting VMs in the ASR 5500VPC-DIVPC-SI Instance

You can create a script that with reset, stop, start or delete one or more VMs in the ASR 5500VPC-DIVPC-SI instance using the hypervisor.

KVM

You can create a KVM script that runs a series of commands to reset, stop, start, or delete specified VMs.

The actual script will contain command sequences appropriate to the installation requirements of the ASR 5500VPC-DIVPC-SI instance as deployed in your site.

Script Commands

Available commands include:

```
 reset
for CARD in {01,02}-cf {03,04,05,06...32}-sf do
  VMNAME={NAME}{INST}-{CARD}
  virsh reset {VMNAME}
done

 stop
for CARD in {01,02}-cf {03,04,05,06...32}-sf do
  VMNAME={NAME}{INST}-{CARD}
  virsh destroy {VMNAME}
done

 start
for CARD in {01,02}-cf {03,04,05,06...32}-sf do
  VMNAME={NAME}{INST}-{CARD}
  virsh start {VMNAME}
done

 delete
for CARD in {01,02}-cf {03,04,05,06...32}-sf do
  VMNAME={NAME}{INST}-{CARD}
  virsh undefine {VMNAME}
done
```

Sample Script

A sample script is provided below:

```
cat ./vm-start

 start local
for CARD in {01,02}-cf {03,04,05,06,17,18,19,20}-sf do
  VMNAME=BLADE5-DI-{CARD}
  virsh start {VMNAME}
done

 start remote
ssh testbed3 '/home/luser/vm-start-BLADE5-DI'

roottestc1b5:/home/luser cat ./vm-stop

 reset local
for CARD in {01,02}-cf {03,04,05,06,17,18,19,20}-sf do
  VMNAME=BLADE5-DI-{CARD}
  virsh destroy {VMNAME}
done

 reset remote
ssh testbed3 '/home/luser/vm-stop-BLADE5-DI'
```

VMware ESXi

VSphere GUI

To reset VMs in a ASR 5500VPC-DIVPC-SI instance you can use the vSphere GUI.

Procedure

Step 1 From vSphere select the host for the target ASR 5500VPC-DIVPC-SI VMs.

Step 2 Select **Power Off** from the Commands list to shut down the VM(s).

Step 3 Select **Power On** from the Commands list to start the VM(s).

Step 4 If multiple vSphere hosts are used to provision the VMs in this ASR 5500VPC-DIVPC-SI instance, repeat steps 2 and 3 for each host.

PowerShell Script

You can also create a PowerShell script that uses the commands described in XREF to restart target VMs.