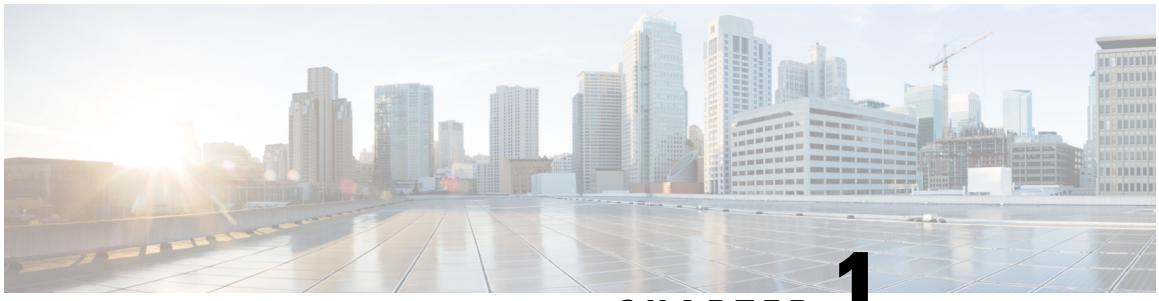
System Setup and Software Installation Guide for Cisco ASR 9000 Series Routers, IOS XR Release 7.5.x

First Published: 2021-11-30

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS



CHAPTER 1

New and Changed Feature Information

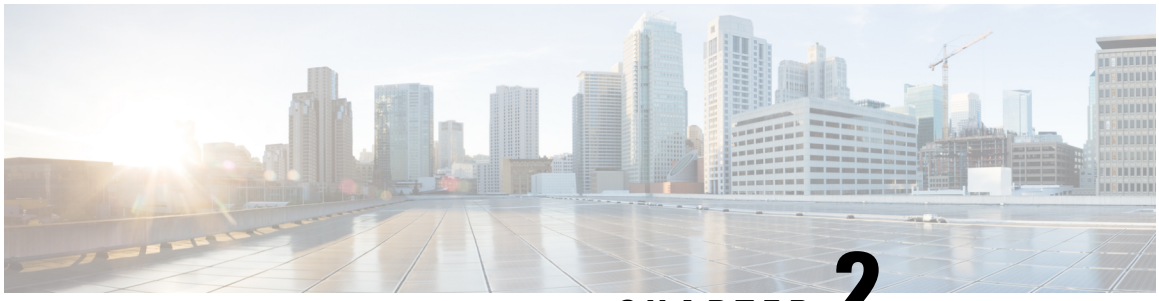
This table summarizes the new and changed feature information for the *System Setup and Software Installation Guide for Cisco ASR 9000 Series Routers* .

New and Changed System Setup Features

Feature	Description	Changed in Release
View Features and Capabilities Supported on a Platform	This functionality displays a list of supported and unsupported features and their capabilities in a release for your router. With this feature, you are better equipped to plan your network configuration with features annotated for their support information. This feature introduces the **show features** command.	Release 7.5.2
View Supported Software Upgrade or Downgrade Versions, on page 45	You can determine whether a software version can be upgraded or downgraded to another version using this functionality. Before an actual upgrade or downgrade process, you can also view the hardware or software limitations that could cause the upgrade or downgrade to fail. This feature helps you plan successful software upgrades or downgrades. This feature introduces the **show install upgrade-matrix** command.	Release 7.5.1
Check Integrity of Golden ISO (GISO) Files	This feature provides an option to verify the integrity of files in GISO using md5sum value.	Release 7.5.1

Feature	Description	Changed in Release
Enhanced Golden ISO Build Tool	This enhancement provides you with the flexibility to use the `gisobuild.py` tool to build GISO images using Cisco IOS XR software commands,YAML-based template file, or Docker capability to suit your customized install requirements. When you build a GISO, you can also specify Zero Touch Provisioning (ZTP) initialization file, script initialization file, Cisco IOS XR configuration file, and SMUs in addition to using the base image and optional RPMs to automatically provision the router.	Release 7.5.1

Stream Telemetry Data for Install Operations	This feature allows you to stream MDT data—both cadence-driven telemetry (CDT) and event-driven telemetry (EDT) data, for changes detected during install operation.
	With this feature, you can stream data from the following sensor paths:
	`Cisco-IOS-XR-install-oper:install/request`
	`Cisco-IOS-XR-install-oper:install/packages/committed/summary`
	`Cisco-IOS-XR-install-oper:install/packages/active/summary`
	`Cisco-IOS-XR-install-oper:install/version`
		Release 7.5.1



CHAPTER 2

Cisco ASR 9000 System Features

The topics covered in this chapter are:

Cisco ASR 9000 Product Overview

The Cisco ASR 9000 series routers are next-generation edge access routers that are optimized for service provider applications. These routers are designed to fulfill various roles in:

- Layer 2 and Layer 3 Ethernet aggregation
- Subscriber-aware broadband aggregation

The Cisco ASR 9000 series routers meet carrier-class requirements for redundancy, availability, packaging, power, and other requirements traditional to the service provider.

The Cisco ASR 9000 series consists of the following routers:

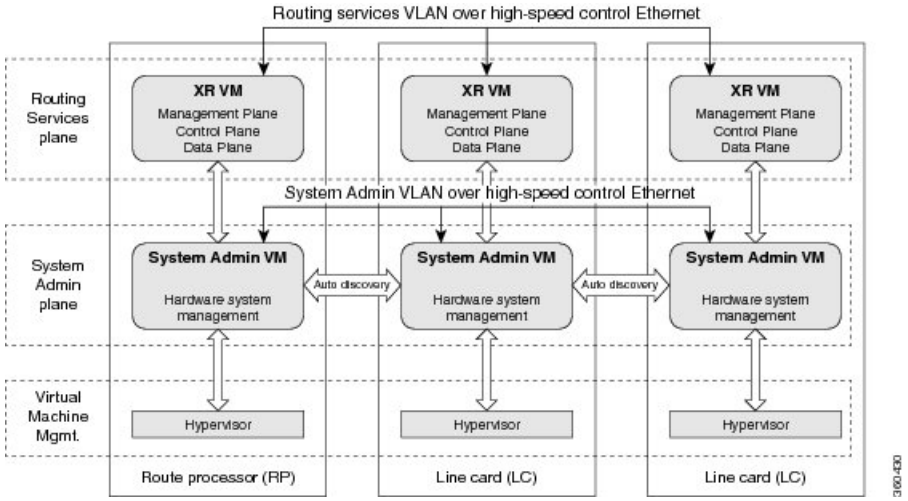
- Cisco ASR 9001 Router (32-bit)
- Cisco ASR 9001-S Router
- Cisco ASR 9006 Router
- Cisco ASR 9010 Router
- Cisco ASR 9901 Router
- Cisco ASR 9904 Router
- Cisco ASR 9906 Router
- Cisco ASR 9910 Router
- Cisco ASR 9912 Router
- Cisco ASR 9922 Router

Virtual Machine based Routing and System Administration

On the Cisco ASR 9000 series router running 64-bit IOS XR, the routing functions and the System Administration functions are run on separate virtual machines (VMs) over a Linux host operating system. The VMs simulate individual physical computing environments over a common hardware. Available hardware resources like processor, memory, hard disk, and so on, are virtualized and allocated to individual virtual machines by the hypervisor.

The VM topology on the Cisco ASR 9000 series router running 64-bit IOS XR is shown in this figure.

Figure 1: Virtualized IOS XR on Cisco ASR 9000 Series Router



Implementation of Virtualized IOS XR on Cisco ASR 9000 Series Router

- The hypervisor creates and manages individual VM environments.
- On every route processor (RP) there are two VMs; one for system administration (System Admin VM) and one for managing the routing functions (XR VM).
- The two VMs on each node operate on their respective planes. On each plane, the VMs are connected to each other using a dedicated VLAN over a high-speed Control Ethernet connection.
- The System Admin VMs can detect each other's presence by auto discovery and thus maintain complete system awareness.

To access the XR VM, connect to the XR VM console port on the RP. To access the System Admin VM, in the XR VM CLI, execute the **admin** command.

Note In 32-bit IOS XR OS, the management interfaces are available from XR VM. In 64-bit IOS XR OS, the Management ports on the RP/RSP are available as follows:

- MGT LAN 0 is available in XR VM.
- MGT LAN 1 is available in Admin VM.

Advantages of Virtualized IOS XR on the Router

- Faster boot time—Because the System Admin functions are on a dedicated VM, the boot time is considerably reduced.
- Independent upgrades—Software packages can be independently installed on the System Admin VM and the XR VM, resulting in minimal system downtime.
- Self-starting VMs—Both the System Admin VM and the XR VM are automatically launched during router boot-up without any user intervention. They have a default set-up that is ready for use.
- System redundancy—In spite of their interconnectivity, there is also a level of isolation between the VMs. Therefore, if a particular VM experiences any issues, it does not affect the functioning of other VMs.

Command Modes

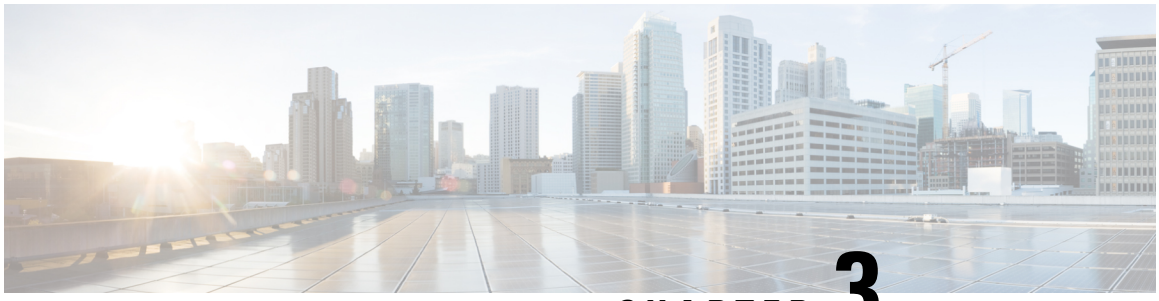
This table lists the command modes:

Command Mode	Description
XR VM Execution Mode	Run commands on the XR VM to display the operational state of the router. Example: `RP/0/RP0/CPU0:router#`

XR VM Global Configuration	Perform security, routing, and other XR feature configurations on the XR VM.
	Example:
	`RP/0/RP0/CPU0:router#configure`
	`RP/0/RP0/CPU0:router(config)#`

System Admin VM Execution Mode	Run commands on the System Admin VM to display and monitor the operational state of the router hardware. The chassis or individual hardware modules can be reloaded from this mode.
	Example:
	`RP/0/RP0/CPU0:router#admin`
	`sysadmin-vm:0_RP0#`

System Admin VM Configuration Mode	Run configuration commands on the System Admin VM to manage and operate the hardware modules of the entire chassis.
	Example:
	`RP/0/RP0/CPU0:router#admin`
	`sysadmin-vm:0_RP0#config`
	`sysadmin-vm:0_RP0(config)#`



CHAPTER 3

Bring-up the Router

After installing the hardware, boot the router. Connect to the XR console port and power on the router. The router completes the boot process using the pre-installed operating system (OS) image. If no image is available within the router, the router can be booted using PXE boot or an external bootable USB drive.

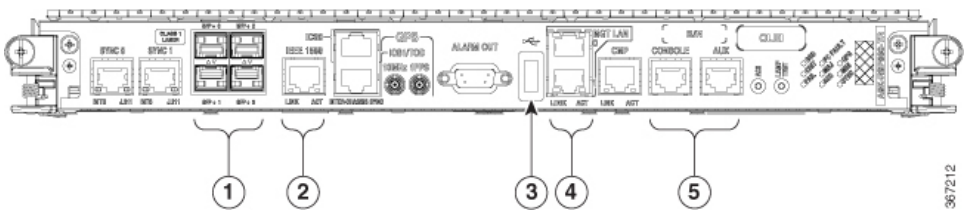
After booting is complete, create the root username and password, and then use it to log on to the XR console and get the router prompt. The first user created in XR console is synchronized to the System Admin console. From the XR console, access the System Admin console to configure system administration settings.

For more information about completing the hardware installation, see Cisco ASR 9000 Series Aggregation Services Router Hardware Installation Guide.

In a large-scale environment, to provision routers remotely without any manually intervention, we recommend you to use Zero Touch Provisioning (ZTP). See Deploy Router Using Classic ZTP, on page 91.

Boot the Router

Use the console port on the Route Processor (RP) to connect to a new router. The console port connect to the XR console by default. If necessary, subsequent connections can be established through the management port, after it is configured.



1	SFP/SFP+ ports
2	Service LAN port

3	External USB port
4	Management LAN ports
5	Console and Auxiliary (AUX) ports

Step 1 Connect a terminal to the console port of the RP.

Step 2 Start the terminal emulation program on your workstation.

- For modular chassis RP, the console settings are baud rate 9600 bps, no parity, 1 stop bits and 8 data bits
- For fixed chassis, the console settings are baud rate 115200 bps, no parity, 1 stop bits and 8 data bits.

The baud rate is set by default and cannot be changed.

For chassis with RSP4, RP2 cards, the console settings are baud rate 9600 bps, no parity, 1 stop bits and 8 data bits. The user can change this baud rate. For next generation RP3, RSP5 cards, the console settings are baud rate 115200 bps, no parity, 1 stop bits and 8 data bits.

Step 3 Power on the router.

Connect the power cord to Power Entry Module (PEM) and the router boots up. The boot process details are displayed on the console screen of the terminal emulation program.

Step 4 Press **Enter**.

The boot process is complete when the system prompts to enter the root-system username. If the prompt does not appear, wait for a while to give the router more time to complete the initial boot procedure, then press **Enter**.

Important If the boot process fails, it may be because the preinstalled image on the router is corrupt. In this case, the router can be booted using an external bootable USB drive.

Note We recommended that you check the `md5sum` of the image after copying from source location to the server from where router boots up with new version. This ensures that if `md5sum` mismatch is observed, you can remove the corrupted file and ensure that a working copy of the image file is available for setup to begin.

What to do next

Specify the root username and password.

Boot the Router Using USB

The bootable USB drive is used to re-image the router for the purpose of system upgrade, password recovery or boot the router in case of boot failure. The USB on router is mounted as disk 2.

Before you begin

Ensure you have completed the following prerequisites:

- You have access to a USB drive with a storage capacity that is between 8GB (min) and 32 GB (max). USB 2.0 and USB 3.0 are supported.
- Copy the compressed boot file from the Software Download Center to your local machine. The file name for the compressed boot file is in the format asr9k-x64-usb_boot-<release_number>.zip.

Step 1 Create a bootable USB drive.

Note The content of the zipped file ("EFI" and "boot" directories) should be extracted directly into root of the USB drive. If the unzipping application places the extracted files in a new folder, move the "EFI" and "boot" directories to root of the USB drive.

a) Connect the USB drive to your local machine and format it with FAT32 or MS-DOS file system using the Windows Operating System or Apple MAC Disk Utility.
b) Copy the `asr9k-x64-usb_boot-<release_number>.zip` compressed boot file to the USB drive.
c) Verify that the copy operation is successful. To verify, compare the file size at source and destination. Additionally, verify the MD5 checksum value.
d) Extract the content of the compressed boot file by unzipping it inside the USB drive. This converts the USB drive to a bootable drive.
e) Eject the USB drive from your local machine.

Step 2 Insert the USB on the active RP, and reload or reset the power of the router.

Note Use this procedure only on active RP; the standby RP must either be removed from the chassis, or stopped at the boot menu. After the active RP is installed with images from USB, boot the standby RP.

Step 3 On active XR console, press `CTRL-C` to view BIOS menu. From the menu, select `IOS-XR 64 bit Local boot using front panel USB media`.

```
Got EMT Mode as Disk Boot
Set OS type None, Received OS type=0
Got Boot Mode as Disk Boot

Booting IOS-XR 64 bit Boot previously installed image - Press Ctrl-c to stop
 .
Please select the operating system and the boot device:
     1) Boot to ROMMON
     2) IOS-XR 64 bit Boot previously installed image
     3) IOS-XR 64 bit Mgmt Network boot using DHCP server
     4) IOS-XR 64 bit Mgmt Network boot using local settings (iPXE)
     (Press 'p' for more option)
Selection [1/2/3/4]: p
Please select the operating system and the boot device:
     1) Boot to ROMMON
     2) IOS-XR 64 bit Boot previously installed image
     3) IOS-XR 64 bit Mgmt Network boot using DHCP server
     4) IOS-XR 64 bit Mgmt Network boot using local settings (iPXE)
     5) IOS-XR 64 bit Internal network boot from RSP/RP
     6) IOS-XR 64 bit Local boot using embedded USB media
     7) IOS-XR 64 bit Local boot using front panel USB media
     8) Change baud rate and continue booting
Selection [1/2/3/4/5/6/7/8]: 7
Selected IOS-XR 64 bit Local boot using front panel USB media, Continue ? Y/N: y

Set CBC OS type IOS-XR 64 bit, EMT USB Boot to CBC
Sending boot success notification

Selected boot option - EFI USB Device 1 (SanDisk Cruzer)
Verifying image signature...
Image signature verified successfully
Image Verification Passed
```

522185

If active and standby RPs are not stopped at the boot menu, the previously used boot option is used. If the system is inactive in the boot menu for 30 minutes, the system resets automatically.

Step 4 If standby RP is present and it was stopped in step 2, boot the standby RP after the active RP starts to boot. From the boot options select `IOS-XR 64 bit Internal network boot from RSP/RP`.

Example:

```
Please select the operating system and the boot device:
     1) IOS-XR (32 bit Classic XR)
     2) IOS-XR 64 bit Boot previously installed image
     3) IOS-XR 64 bit Mgmt Network boot using DHCP server
     4) IOS-XR 64 bit Mgmt Network boot using local settings (iPXE)
     5) IOS-XR 64 bit Internal network boot from RSP/RP
     6) IOS-XR 64 bit Local boot using embedded USB media
     7) IOS-XR 64 bit Local boot using front panel USB media
Selection [1/2/3/4/5/6/7]:
```

Select option 5 and proceed with the boot up. After the router boots up, specify the root username and password.

Boot the Router Using iPXE

iPXE is a pre-boot execution environment that is included in the network card of the management interfaces and works at the system firmware (UEFI) level of the router. iPXE is used to re-image the system, and boot

the router in case of boot failure or in the absence of a valid bootable partition. iPXE downloads the ISO image, proceeds with the installation of the image, and bootstraps within the new installation.

iPXE acts as a boot loader and provides the flexibility to choose the image that the system will boot based on the Platform Identifier (PID), the Serial Number, or the management mac-address. iPXE must be defined in the DHCP server configuration file.

Note PID and serial number is supported only if iPXE is invoked using the command (admin) hw-module location all bootmedia network reload all. If iPXE is selected manually from BIOS, PID and serial number is not supported.

Note **Cisco ASR 9901** — By default, iPXE uses the previous attempted boot method on the next reload. If the Network option was previously used, the iPXE register will be set to 1 (IPXE_PREF=1). To boot an Cisco ASR 9901 router via DHCP on the next reload, you must set the IPXE_PREF settings to 0 (IPXE_PREF=0).

From the system admin console, enter the **run chvrf 0 ssh rp0_admin /opt/cisco/calvados/bin/nvram_dump -s IPXE_PREF=0** command twice. After entering this command the first time, the host is added to the known list of hosts.

```
sysadmin-vm:0_RP0# run chvrf 0 ssh rp0_admin /opt/cisco/calvados/bin/nvram_dump -s IPXE_PREF=0
Sat May  2  10:39:52.740 UTC+00:00
Warning: Permanently added 'rp0_admin' (ECDSA) to the list of known hosts.
sysadmin-vm:0_RP0# run chvrf 0 ssh rp0_admin /opt/cisco/calvados/bin/nvram_dump -s IPXE_PREF=0
Sat May  2  10:39:54.995 UTC+00:00
sysadmin-vm:0_RP0# hw-module location all bootmedia network
```

iPXE boot can be performed during the following scenarios:

- migration from 32-bit to 64-bit using migration script
- recover password
- boot-up failure with 64-bit image

Before you begin

Take a backup of configuration to a TFP or FTP path to load the configuration back after the iPXE boot.

Step 1 Login to the system admin console.

Example:

```
sysadmin-vm:0_RSP0# hw-module location all reload
Tue Mar  6  08:12:47.605 UTC
Reload hardware module ? [no,yes] yes
result Card graceful reload request on all acknowledged.
sysadmin-vm:0_RSP0#
```

Step 2 If the router is unable to boot, press Ctrl +C to stop the boot process when the following information is displayed.

Note Use this procedure only on active RP; the standby RP must either be removed from the chassis, or stopped at the boot menu. After the active RP is installed with images from iPXE boot, boot the standby RP.

Example:

```
System Bootstrap, Version 10.57 [ASR9K x86 ROMMON],
Copyright (c) 1994-2018 by Cisco Systems, Inc.
Compiled on Mon 01/09/2017 17:15:01.98
BOARD_TYPE             : 0x100317
Rommon                 : 10.57 (Primary)
Board Revision         : 4
PCH EEPROM             : 3.4
IPU FPGA(PL)           : 0.40.0 (Backup)
IPU INIT(HW_FPD)       : 0.30.0
IPU FSBL(BOOT.BIN)     : 1.19.0
IPU LINUX(IMAGE.FPD)   : 1.21.0
OPTIMUS FPGA           : 0.12.0
OMEGA FPGA             : 0.13.0
ALPHA FPGA             : 0.14.0
CHA FPGA               : 0.5.1
CBC0                   : Part 1=34.38, Part 2=34.38, Act Part=2
Product Number         : A9K-RSP880-SE
Chassis                : ASR-9904-AC
Chassis Serial Number  : FOX1936GBDD
Slot Number            : 1
Pxe Mac Address LAN 0  : 70:e4:22:06:13:40
Pxe Mac Address LAN 1  : 70:e4:22:06:13:41
=========================================================
Got EMT Mode as 3
Got Boot Mode as 0
Booting IOS-XR (32 bit Classic XR) - Press Ctrl-c to stop
```

Step 3 Choose option 4 for iPXE boot.

Example:

```
Please select the operating system and the boot device:
     1) IOS-XR (32 bit Classic XR)
     2) IOS-XR 64 bit Boot previously installed image
     3) IOS-XR 64 bit Mgmt Network boot using DHCP server
     4) IOS-XR 64 bit Mgmt Network boot using local settings (iPXE)
     5) IOS-XR 64 bit Internal network boot from RSP/RP
     6) IOS-XR 64 bit Local boot using embedded USB media
     7) IOS-XR 64 bit Local boot using front panel USB media
Selection [1/2/3/4/5/6/7]:
```

Step 4 Manually update iPXE ROMMON details before booting using FTP or TFTP.

Example:

```
iPXE>set cisco/cisco-server-url:string tftp://<path>/asr9k-mini-x64.iso
iPXE>set cisco/cisco-ipv4-address:string 1.3.24.202
iPXE>set cisco/cisco-netmask-address:str 255.255.0.0
iPXE>set cisco/cisco-gateway-address:str 1.3.0.1
```

Step 5 Open the connected management port (0/1).

Example:

```
iPXE>ifclose net0
iPXE>ifclose net1
iPXE>ifopen net1
```

where net0 and net1 represents management port0 and port1 respectively.

Step 6 Boot the required image from FTP or TFTP location.

Example:

```
iPXE>
iPXE> ifopen net0:
iPXE> boot tftp://<path>/asr9k-mini-x64-<release-number>.iso
tftp://<path>/asr9k-mini-x64-<release-number>.iso...  0%
Booting iso-image@0x83c525000(1135456256), bzImage@0x83c55f000(4526671)

.......BIOS CODE SIGN ENTRY ...
```

Step 7 After the active RP is up and running, boot the standby RP. From the boot options select `IOS-XR 64 bit Internal network boot from RSP/RP`.

Example:

```
Please select the operating system and the boot device:
     1) IOS-XR (32 bit Classic XR)
     2) IOS-XR 64 bit Boot previously installed image
     3) IOS-XR 64 bit Mgmt Network boot using DHCP server
     4) IOS-XR 64 bit Mgmt Network boot using local settings (iPXE)
     5) IOS-XR 64 bit Internal network boot from RSP/RP
     6) IOS-XR 64 bit Local boot using embedded USB media
     7) IOS-XR 64 bit Local boot using front panel USB media
Selection [1/2/3/4/5/6/7]:
```

Setup Root User Credentials

When you boot the router for the first time, the system prompts you to configure root credentials (username and password). These credentials have been set up for the root user on the XR console (root-lr), the System Admin VM (root-system), and for disaster recovery purposes.

Step 1 **Enter root-system username:** *username*

Enter the username of the root user. The character limit is 1023. In this example, the name of the root user is "root".

Important The specified username is mapped to the "root-lr" group on the XR console. It is also mapped as the "root-system" user on the System Admin console.

When starting the router for the first time, or after resetting the router's operating system to its default state, the router does not have any user configuration. In such cases, the router prompts you to specify the "root-system username". However, if the router has been configured previously, the router prompts you to enter the "username", as described in Step 4.

Step 2 **Enter secret:** *password*

Enter the password for the root user. The character range of the password is from 6 through 253 characters. The password that you type is not displayed on the CLI for security reasons.

The root-system username and password must be safeguarded as they have superuser privileges. They are used to access the complete router configuration.

Step 3 **Enter secret again:** *password*

Reenter the password for the root-system user. The password that you type is not displayed on the CLI for security reasons.

Step 4 **Username:** *username*

Enter the root-system username to login to the XR VM console.

Step 5 **Password:** *password*

Enter the password of the root-system user. The correct password displays the router prompt. You are now logged into the XR VM console.

Step 6 (Optional) **show run username**

Displays user details.

```
username root
 group root-lr
 group cisco-support
 secret 5 $1$NBg7$fHs1inKPZVvzqxMv775UE/
!
```

What to do next

- Configure routing functions from the XR console.
- Configure system administration settings from the System Admin prompt. The System Admin prompt is displayed on accessing the System Admin console. For details on how to get the System Admin prompt, see Access the System Admin Console, on page 14.

Access the System Admin Console

You must log in to the System Admin console through the XR console to perform all system administration and hardware management setup.

Step 1 Log in to the XR console as the root user.

Step 2 (Optional) Disable the login banner on console port when accessing the System Admin mode from XR mode.

a) **configure**

b) **service sysadmin-login-banner disable**

Example:

```
RP/0/RSP0/CPU0:router(config)#service sysadmin-login-banner disable
```

Disable the login banner on console port in System Admin mode.

c) **commit**

d) **end**

Step 3 **admin**

Example:

The login banner is enabled by default. The following example shows the command output with the login banner enabled:

```
RP/0/RSP0/CPU0:router#admin

Mon May 22 06:57:29.350 UTC
```

```
root connected from 127.0.0.1 using console on host
sysadmin-vm:0_RP0# exit
Mon May  22 06:57:32.360 UTC
```

The following example shows the command output with the login banner disabled:

```
RP/0/RP0/CPU0:router#admin
Thu Mar 01:07:14.509 UTC
sysadmin-vm:0_RP0# exit
```

Step 4 (Optional) **exit**

Return to the XR mode from the System Admin mode.

Configure the Management Port

To use the Management port for system management and remote communication, you must configure an IP address and a subnet mask for the management ethernet interface. To communicate with devices on other networks (such as remote management stations or TFTP servers), you need to configure a default (static) route for the router.

Before you begin

- Consult your network administrator or system planner to procure IP addresses and a subnet mask for the management interface.
- Physical port Ethernet 0 and Ethernet 1 on RP are the management ports. Ensure that the port is connected to management network.

SUMMARY STEPS

1. **configure**
2. **interface MgmtEth** *rack/slot/port*
3. **ipv4 address** *ipv4-address subnet-mask*
4. **ipv4 address** *ipv4 virtual address subnet-mask*
5. **no shutdown**
6. **exit**
7. **router static address-family ipv4 unicast** *0.0.0.0/0 default-gateway*
8. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **interface MgmtEth** *rack/slot/port*

Example:

```
RP/0/RSP0/CPU0:router(config)#interface mgmtEth 0/RP0/CPU0/0
```

Enters interface configuration mode for the management interface of the primary RP.

Step 3 **ipv4 address** *ipv4-address subnet-mask*

Example:

```
RP/0/RSP0/CPU0:router(config-if)#ipv4 address 10.1.1.1/8
```

Assigns an IP address and a subnet mask to the interface.

Step 4 **ipv4 address** *ipv4 virtual address subnet-mask*

Example:

```
RP/0/RSP0/CPU0:router(config-if)#ipv4 address 1.70.31.160 255.255.0.0
```

Assigns a virtual IP address and a subnet mask to the interface.

Step 5 **no shutdown**

Example:

```
RP/0/RSP0/CPU0:router(config-if)#no shutdown
```

Places the interface in an "up" state.

Step 6 **exit**

Example:

```
RP/0/RSP0/CPU0:router(config-if)#exit
```

Exits the Management interface configuration mode.

Step 7 **router static address-family ipv4 unicast** *0.0.0.0/0 default-gateway*

Example:

```
RP/0/RSP0/CPU0:router(config)#router static address-family ipv4 unicast 0.0.0.0/0 12.25.0.1
```

Specifies the IP address of the default-gateway to configure a static route; this is to be used for communications with devices on other networks.

Step 8 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

What to do next

Connect to the management port to the ethernet network. With a terminal emulation program, establish a SSH or telnet connection to the management interface port using its IP address. Before establishing a telnet session, use the **telnet ipv4|ipv6 server max-servers** command in the XR Config mode, to set number of allowable telnet sessions to the router.

Perform Clock Synchronization with NTP Server

There are independent system clocks for the XR console and the System Admin console. To ensure that these clocks do not deviate from true time, they need to be synchronized with the clock of a NTP server. In this task you will configure a NTP server for the XR console. After the XR console clock is synchronized, the System Admin console clock will automatically synchronize with the XR console clock.

Before you begin

Configure and connect to the management port.

Step 1 **configure**

Example:

```
RP/0/RSP0/CPU0:router# configure
```

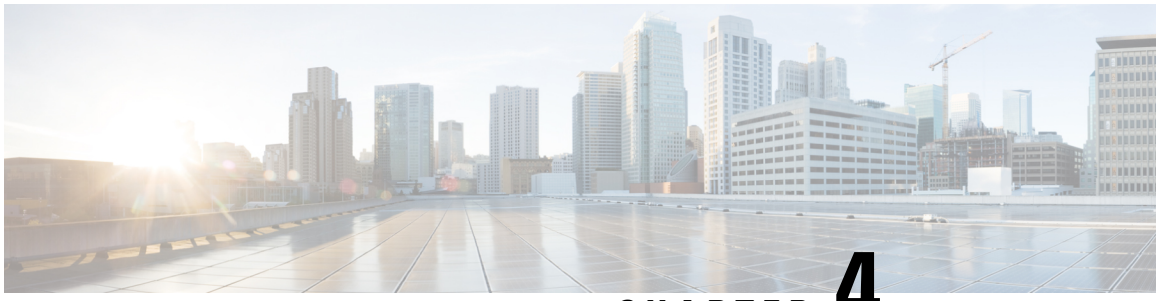
Enters global configuration mode.

Step 2 **ntp server** *server_address*

Example:

```
RP/0/RSP0/CPU0:router(config)#ntp server 64.90.182.55
```

The XR console clock is configured to be synchronized with the specified sever.



CHAPTER 4

Perform Preliminary Checks

After successfully logging into the console, you must perform some preliminary checks to verify the default setup. If any setup issue is detected when these checks are performed, take corrective action before making further configurations. These preliminary checks are:

Verify Software Version

The router is shipped with the Cisco IOS XR software pre-installed. Verify that the latest version of the software is installed. If a newer version is available, perform a system upgrade. This will install the newer version of the software and provide the latest feature set on the router.

Perform this task to verify the version of Cisco IOS XR software running on the router.

SUMMARY STEPS

1. **show version**

DETAILED STEPS

show version

Example:

```
RP/0/RSP0/CPU0:router# show version
```

Displays the version of the various software components installed on the router. The result includes the version of Cisco IOS XR software and its various components.

Example

What to do next

Verify the result to ascertain whether a system upgrade or additional package installation is required. If that is required, refer to the tasks in the chapter Perform System Upgrade and Install Feature Packages.

Verify Active VMs

On the router both the XR VM and the System Admin VM must be operational. Instances of both VMs should be running on every route processor (RP). Complete this task to verify the VMs are active.

SUMMARY STEPS

1. **show redundancy summary**
2. **admin**
3. **show vm**

DETAILED STEPS

Step 1 **show redundancy summary**

Example:

```
RP/0/RP0:hostname#show redundancy summary
Mon Mar 9 16:32:19.276 IST
Active Node Standby Node
----------- ------------
0/RP0 0/RP1 (Node Ready, NSR: Not Configured)
0/LC0 0/LC1 (Node Ready, NSR: Not Configured)
RP/0/RP0:hostname#
```

Displays the readiness of the VMs.

Step 2 **admin**

Example:

```
RP/0/RSP0/CPU0:router# admin
```

Enters administration EXEC mode.

Step 3 **show vm**

Example:

```
sysadmin-vm:0_RP0#show vm
```

Displays the status of the VMs running on various nodes.

```
sysadmin-vm:0_RP0# sh vm
Mon Mar 9 07:52:06.173 UTC
------ VMs found at location 0/RP0 ------
Id : sysadmin
Status : running
```

```
IP Addr: 192.0.44.1
HB Interval : NA
Last HB Sent: NA
Last HB Rec : NA
-------
Id : default-sdr
Status : running
IP Addr: 192.0.44.4
HB Interval : 0 s 500000000 ns
Last HB Sent: 663743
Last HB Rec : 663743
-------
Id : default-sdr
Status : running
IP Addr: 192.0.44.6
HB Interval : 10 s 0 ns
Last HB Sent: 33183
Last HB Rec : 33183
-------
------ VMs found at location 0/RP1 ------
Id : sysadmin
Status : running
IP Addr: 192.0.88.1
HB Interval : NA
Last HB Sent: NA
Last HB Rec : NA
-------
Id : default-sdr
Status : running
IP Addr: 192.0.88.4
HB Interval : 0 s 500000000 ns
Last HB Sent: 663749
Last HB Rec : 663749
-------
Id : default-sdr
Status : running
IP Addr: 192.0.88.6
HB Interval : 10 s 0 ns
Last HB Sent: 33183
Last HB Rec : 33183
-------
sysadmin-vm:0_RP0#
```

In the above result:

- Id—Name of the VM. "sysadmin" represents System Admin VM; "default-sdr" represents XR VM.
- Status—Status of the VM
- IP Addr—Internal IP address of the VM

If a VM is not running on a node, in the output of the **show vm** command, no output is shown for that node.

What to do next

If the XR VM is not running on a node, try reloading the node. To do so, use the **hw-module location** *node-id* **reload** command in the mode. Also, use the **show sdr** command in the mode to verify that the SDR is running on the node.

Verify Status of Hardware Modules

Hardware modules include RPs, fan trays, and so on. On the router, multiple hardware modules are installed. Perform this task to verify that all hardware modules are installed correctly and are operational.

Before you begin

Ensure that all required hardware modules have been installed on the router.

Verify Firmware Version

The firmware on various hardware components of the router must be compatible with the Cisco IOS XR image installed. Incompatibility might cause the router to malfunction. Complete this task to verify the firmware version.

SUMMARY STEPS

1. **show hw-module fpd**

DETAILED STEPS

show hw-module fpd

Example:

Displays the list of hardware modules detected on the router.

Note This command can be run from both XR VM and System Admin VM modes.

In the above output, some of the significant fields are:

- FPD Device- Name of the hardware component such as FPD, CFP, and so on.
- ATR-Attribute of the hardware component. Some of the attributes are:
  - B- Backup Image
  - S-Secure Image
  - P-Protected Image
- Status- Upgrade status of the firmware. The different states are:
  - CURRENT-The firmware version is the latest version.
  - READY-The firmware of the FPD is ready for an upgrade.
  - NOT READY-The firmware of the FPD is not ready for an upgrade.
  - NEED UPGD-A newer firmware version is available in the installed image. It is recommended that an upgrade be performed.
  - RLOAD REQ-The upgrade has been completed, and the ISO image requires a reload.

- UPGD DONE-The firmware upgrade is successful.
- UPGD FAIL- The firmware upgrade has failed.
- BACK IMG-The firmware is corrupted. Reinstall the firmware.
- UPGD SKIP-The upgrade has been skipped because the installed firmware version is higher than the one available in the image.

- Running- Current version of the firmware running on the FPD.

What to do next

- Upgrade the required firmware by using the **upgrade hw-module location all fpd** command in the EXEC mode. For the FPD upgrade to take effect, the router needs a power cycle.
- It is recommended to upgrade all FPGAs on a given node using the **upgrade hw-module fpd all location** {**all** | *node-id*} command. Do not upgrade the FPGA on a node using the **upgrade hw-module fpd** <*individual-fpd*> **location** {**all** | *node-id*} as it may cause errors in booting the card.
- If required, turn on the auto fpd upgrade function. To do so, use the **fpd auto-upgrade enable** command in the XR configuration [**(config)#**] mode. After it is enabled, if there are new FPD binaries present in the image being installed on the router, FPDs are automatically upgraded during the system upgrade operation.

Verify SDR Information

Secure domain routers (SDRs) divide a single physical system into multiple logically-separated routers. SDRs are also known as logical routers (LRs). On the router, only one SDR is supported. This SDR is termed the default-sdr. Every router is shipped with the default-sdr, which owns all RPs installed in the routing system. An instance of this SDR runs on line cards and route processors. Complete this task to verify the details of the SDR instances.

SUMMARY STEPS

1. **admin**
2. **show sdr**

DETAILED STEPS

Step 1 **admin**

Example:

```
RP/0/RSP0/CPU0:router# admin
```

Enters administration EXEC mode.

Step 2 **show sdr**

Example:

```
sysadmin-vm:0_RP0# show sdr
```

Displays the SDR information for every node.

```
sysadmin-vm:0_RP0# show sdr

sdr default-sdr
 location 0/0/VM1
  sdr-id             2
  IP Address of VM   192.0.4.3
  MAC address of VM  A4:6C:2A:2B:AA:A6
  VM State           RUNNING
  start-time         2015-12-03T15:38:38.74514+00:00
  Last Reload Reason "SMU:Reboot triggered by install"
  Reboot Count       2
 location 0/1/VM1
  sdr-id             2
  IP Address of VM   192.0.8.3
  MAC address of VM  B0:AA:77:E7:5E:DA
  VM State           RUNNING
  start-time         2015-12-03T15:38:39.730036+00:00
  Last Reload Reason "SMU:Reboot triggered by install"
  Reboot Count       2
 location 0/2/VM1
  sdr-id             2
  IP Address of VM   192.0.12.3
  MAC address of VM  B0:AA:77:E7:67:34
  VM State           RUNNING
  start-time         2015-12-03T15:38:38.886947+00:00
  Last Reload Reason "SMU:Reboot triggered by install"
  Reboot Count       2
 location 0/3/VM1
  sdr-id             2
  IP Address of VM   192.0.16.3
  MAC address of VM  B0:AA:77:E7:58:86
  VM State           RUNNING
  start-time         2015-12-03T15:38:40.391205+00:00
  Last Reload Reason "SMU:Reboot triggered by install"
  Reboot Count       2
 location 0/4/VM1
  sdr-id             2
  IP Address of VM   192.0.20.3
  MAC address of VM  B0:AA:77:E7:46:C2
  VM State           RUNNING
  start-time         2015-12-03T15:38:39.84469+00:00
  Last Reload Reason "SMU:Reboot triggered by install"
  Reboot Count       2
 location 0/5/VM1
  sdr-id             2
  IP Address of VM   192.0.24.3
  MAC address of VM  B0:AA:77:E7:84:40
  VM State           RUNNING
  start-time         2015-12-04T03:48:24.017443+00:00
  Last Reload Reason "VM_REQUESTED_UNGRACEFUL_RELOAD:Headless SDR"
  Reboot Count       3
 location 0/6/VM1
  sdr-id             2
  IP Address of VM   192.0.28.3
  MAC address of VM  B0:AA:77:E7:55:FE
  VM State           RUNNING
  start-time         2015-12-03T15:38:38.74753+00:00
  Last Reload Reason "SMU:Reboot triggered by install"
  Reboot Count       2
```

```
location 0/7/VM1
 sdr-id             2
 IP Address of VM   192.0.32.3
 MAC address of VM  B0:AA:77:E7:60:C6
 VM State           RUNNING
 start-time         2015-12-03T15:38:38.691481+00:00
 Last Reload Reason "SMU:Reboot triggered by install"
 Reboot Count       2
location 0/RP0/VM1
 sdr-id             2
 IP Address of VM   192.0.108.4
 MAC address of VM  10:05:CA:D7:FE:6F
 VM State           RUNNING
 start-time         2015-12-04T07:03:04.549294+00:00
 Last Reload Reason CARD_SHUTDOWN
 Reboot Count       1
location 0/RP1/VM1
 sdr-id             2
 IP Address of VM   192.0.112.4
 MAC address of VM  10:05:CA:D8:3F:43
 VM State           RUNNING
 start-time         2015-12-04T09:21:42.083046+00:00
 Last Reload Reason CARD_SHUTDOWN
 Reboot Count       1
```

For a functional SDR, the VM State is "RUNNING". If the SDR is not running on a node, no output is shown in the result, for that location.

What to do next

If you find SDR is not running on a node, try reloading the node. To do that, use the **hw-module location** *node-id* **reload** command in the mode.

Verify Interface Status

After the router has booted, all available interfaces must be discovered by the system. If interfaces are not discovered, it might indicate a malfunction in the unit. Complete this task to view the number of discovered interfaces.

SUMMARY STEPS

1. **show ipv4 interface summary**

DETAILED STEPS

show ipv4 interface summary

Example:

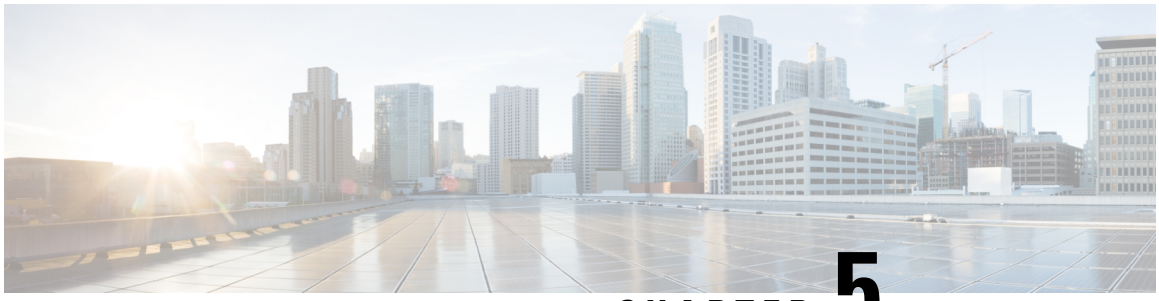
```
RP/0/RSP0/CPU0:router#show ipv4 interface summary
```

When a router is turned on for the first time, all interfaces are in the 'unassigned' state. Verify that the total number of interfaces displayed in the result matches with the actual number of interfaces present on the router.

In the above result:

- Assigned— An IP address is assigned to the interface.
- Unnumbered— Interface which has borrowed an IP address already configured on one of the other interfaces of the router.
- Unassigned—No IP address is assigned to the interface.

You can also use the **show interfaces brief** and **show interfaces summary** commands in the mode to verify the interface status.



CHAPTER 5

Create User Profiles and Assign Privileges

To provide controlled access to the XR and System Admin configurations on the router, user profiles are created with assigned privileges. The privileges are specified using command rules and data rules.

The authentication, authorization, and accounting (aaa) commands are used for the creation of users, groups, command rules, and data rules. The `aaa` commands are also used for changing the disaster-recovery password.

Note You cannot configure the external AAA server and services from the System Admin VM. It can be configured only from the XR VM.

Configure AAA authorization to restrict users from uncontrolled access. If AAA authorization is not configured, the command and data rules associated to the groups that are assigned to the user are bypassed. An IOS-XR user can have full read-write access to the IOS-XR configuration through Network Configuration Protocol (NETCONF), google-defined Remote Procedure Calls (gRPC) or any YANG-based agents. In order to avoid granting uncontrolled access, enable AAA authorization before setting up any configuration.

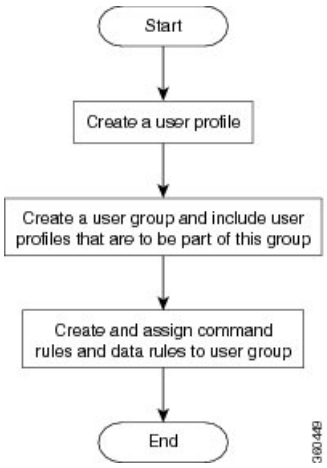
Note If any user on XR is deleted, the local database checks whether there is a first user on System Admin VM.

- If there is a first user, no syncing occurs.
- If there is no first user, then the first user on XR (based on the order of creation) is synced to System Admin VM.
- When a user is added in XR, if there is no user on System Admin mode, then the user is synced to sysadmin-vm. After the synchronization, any changes to the user on XR VM does not synchronize on the System Admin VM.
- A user added on the System Admin VM does not synchronize with XR VM.
- Only the first user or disaster-recovery user created on System Admin VM synchronizes with the host VM.
- Changes to credentials of first user or disaster-recovery user on System Admin VM synchronizes with the host VM.
- The first user or disaster-recovery user deleted on System Admin VM does not synchronize with the host VM. The host VM retains the user.

Users are authenticated using username and password. Authenticated users are entitled to execute commands and access data elements based on the command rules and data rules that are created and applied to user groups. All users who are part of a user group have such access privileges to the system as defined in the command rules and data rules for that user group.

The workflow for creating user profile is represented in this flow chart:

Figure 2: Workflow for Creating User Profiles



Note The root-lr user, created for the XR VM during initial router start-up, is mapped to the root-system user for the System Admin VM. The root-system user has superuser permissions for the System Admin VM and therefore has no access restrictions.

Use the **show run aaa** command in the Config mode to view existing aaa configurations.

The topics covered in this chapter are:

Create User Groups

Create a new user group to associate command rules and data rules with it. The command rules and data rules are enforced on all users that are part of the user group.

For extensive information about creating user groups, task groups, RADIUS and TACACS configurations, see the *Configuring AAA Services* chapter in the *System Security Configuration Guide for Cisco ASR 9000 Series Routers*. For detailed information about commands, syntax and their description, see the *Authentication, Authorization, and Accounting Commands* chapter in the *System Security Command Reference for Cisco ASR 9000 Series Routers*.

Configure User Groups in XR VM

User groups are configured with the command parameters for a set of users, such as task groups. Entering the **usergroup** command accesses the user group configuration submode. Users can remove specific user groups by using the **no** form of the **usergroup** command. Deleting a usergroup that is still referenced in the system results in a warning.

Before you begin

Note Only users associated with the WRITE:AAA task ID can configure user groups. User groups cannot inherit properties from predefined groups, such as owner-sdr.

Step 1 **configure**

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **usergroup** *usergroup-name*

Example:

```
 RP/0/RSP0/CPU0:router(config)# usergroup beta
```

Creates a name for a particular user group and enters user group configuration submode.

- Specific user groups can be removed from the system by specifying the **no** form of the **usergroup** command.

Step 3 **description** *string*

Example:

```
 RP/0/RSP0/CPU0:router(config-ug)#
description this is a sample user group description
```

(Optional) Creates a description of the user group named in Step 2.

Step 4 **inherit usergroup** *usergroup-name*

Example:

```
 RP/0/RSP0/CPU0:router(config-ug)#
inherit usergroup sales
```

- Explicitly defines permissions for the user group.

Step 5 **taskgroup** *taskgroup-name*

Example:

```
 RP/0/RSP0/CPU0:router(config-ug)# taskgroup beta
```

Associates the user group named in Step 2 with the task group named in this step.

- The user group takes on the configuration attributes (task ID list and permissions) already defined for the entered task group.

Step 6 Repeat Step for each task group to be associated with the user group named in Step 2.

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Create a User Group in System Admin VM

Create a user group for the System Admin VM.

The router supports a maximum of 32 user groups.

Before you begin

Create a user profile. See the *Create User* section.

Step 1 **admin**

Example:

```
RP/0/RSP0/CPU0:router# admin
```

Enters administration EXEC mode.

Step 2 **config**

Example:

```
sysadmin-vm:0_RP0#config
```

Enters mode.

Step 3 **aaa authentication groups group** *group_name*

Example:

```
sysadmin-vm:0_RP0(config)#aaa authentication groups group gr1
```

Creates a new user group (if it is not already present) and enters the group configuration mode. In this example, the user group "gr1" is created.

Note By default, the user group "root-system" is created by the system at the time of root user creation. The root user is part of this user group. Users added to this group will get root user permissions.

Step 4 **users** *user_name*

Example:

```
sysadmin-vm:0_RP0(config-group-gr1)#users us1
```

Specify the name of the user that should be part of the user group.

You can specify multiple user names enclosed withing double quotes. For example, **users** "*user1 user2 ...*".

Step 5 **gid** *group_id_value*

Example:

```
sysadmin-vm:0_RP0(config-group-gr1)#gid 50
```

Specify a numeric value. You can enter any 32 bit integer.

Step 6 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

What to do next

- Create command rules.
- Create data rules.

Create Users

You can create new users and include the user in a user group with certain privileges. The router supports a maximum of 1024 user profiles.

Note Users created in the System Admin VM are different from the ones created in XR VM. As a result, the username and password of a System Admin VM user cannot be used to access the XR VM, and vice versa.

XR VM and System Admin VM User Profile Synchronization

Initial User Profile Synchronization: When a user profile is created for the first time within the XR VM, the username and password are synchronized with the System Admin VM, but only if the user does not already exist in the System Admin VM. This initial synchronization ensures consistent user information between the two VMs.

Limitations on Subsequent Changes: However, it is important to note that the System Admin VM does not synchronize subsequent password changes or user deletions made within the XR VM. Consequently, the passwords in the XR VM and the System Admin VM may differ, and user profiles may not be updated in real time to reflect deletions within the XR VM.

User Deleting Handling: Additionally, when a user is deleted within the XR VM, the corresponding user profile in the System Admin VM remains unaffected. In other words, user deletion in the XR VM does not automatically remove the user's profile in the System Admin VM.

For extensive information about creating user groups, task groups, RADIUS and TACACS configurations, see the *Configuring AAA Services* chapter in the *System Security Configuration Guide for Cisco ASR 9000 Series Routers*. For detailed information about commands, syntax and their description, see the *Authentication, Authorization, and Accounting Commands* chapter in the *System Security Command Reference for Cisco ASR 9000 Series Routers*.

Create a User Profile in XR VM

Table 1: Feature History Table

Feature name	Release Information	Feature Description
Enhanced Login Banner Standards	Release 7.3.1	To comply with the US DoD, an option to enable display of login banner is introduced. The login banner provides information such as number of successful and unsuccessful login attempts, time stamp, login method, and so on. The login-history command is introduced.

Each user is identified by a username that is unique across the administrative domain. Each user must be a member of at least one user group. Deleting a user group may orphan the users associated with that group. The AAA server authenticates orphaned users but most commands are not authorized.

For more information about AAA, and creating users, see the *Configuring AAA Services* chapter in the *System Security Configuration Guide for Cisco ASR 9000 Series Routers*. For detailed information about related commands, syntax and their description, see the *Authentication, Authorization, and Accounting Commands* chapter in the *System Security Command Reference for Cisco ASR 9000 Series Routers*.

Step 1 **configure**

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **username** *user-name*

Example:

```
RP/0/RSP0/CPU0:router(config)# username user1
```

Creates a name for a new user (or identifies a current user) and enters username configuration submode.

- The *user-name* argument can be only one word. Spaces and quotation marks are not allowed.

Step 3 Do one of the following:

- **password** {**0** | **7**} *password*
- **secret** {**0** | **5** | **8** | **9** | **10**} *secret*

Example:

```
Router(config-un)# password 0 pwd1
```

or

```
Router(config-un)# secret 0 sec1
```

Specifies a password for the user named in Step 2.

- Use the **secret** command to create a secure login password for the user names specified in Step 2.
- Entering **0** following the **password** command specifies that an unencrypted (clear-text) password follows. Entering **7** following the **password** command specifies that an encrypted password follows.
- For the **secret** command, the following values can be entered:
  - **0** : specifies that a secure unencrypted (clear-text) password follows
  - **5** : specifies that a secure encrypted password follows that uses MD5 hashing algorithm
  - **8** : specifies that Type 8 secret that uses SHA256 hashing algorithm follows
  - **9** : specifies that Type 9 secret that uses SCrypt hashing algorithm follows

    Note The Type 8 and Type 9 secrets are supported on the IOS XR 64-bit operating system starting from Cisco IOS XR Software Release 7.0.1. Prior to this release, it was supported only on the IOS XR 32-bit operating system.

  - **10** : specifies Type 10 secret that uses SHA512 hashing algorithm

    Note
    - Type 10 secret is supported only for Cisco IOS XR 64 bit platform.
    - Backward compatibility issues such as configuration loss, authentication failure, and so on, are expected when you downgrade to lower versions that still use **MD5** or **SHA256** encryption algorithms. If there are any type 10 secrets, convert the **secrets** to type 5 if you are downgrading the system from versions 7.0.1 and above to versions 6.5.3 and above. If you are downgrading the system from versions 7.0.1 and above to versions below 6.5.3, then un-configure all users from the XR-vm and sysadmin-vm before executing install activate. Backward compatibility issue does not occur in Cisco ASR 9000 Series Routers running Cisco IOS XR 32-Bit software because Type 10 secret is not applicable to such routers.
    - In a first user configuration scenario or when you reconfigure a user, the system synchronises only the Type 5 and Type 10 secrets from XR VM to System Admin VM and Host VM. It does not synchronize the Type 8 and Type 9 secrets in such scenarios.

- Type **0** is the default for the **password** and **secret** commands.
- From Cisco IOS XR Software Release 7.0.1 and later, the default hashing type is 10 (SHA512) when clear text secret is configured without choosing the type in the configuration.

Step 4 **group** *group-name*

Example:

```
RP/0/RSP0/CPU0:router(config-un)# group sysadmin
```

Assigns the user named in Step 2 to a user group that has already been defined through the **usergroup** command.

- The user takes on all attributes of the user group, as defined by that user group's association to various task groups.
- Each user must be assigned to at least one user group. A user may belong to multiple user groups.

Step 5 Repeat step 4 for each user group to be associated with the user specified in step 2.

Step 6 (Optional) You can enable the display of the US Department of Defense DOD-approved login banner. The banner is displayed before granting access to devices. The banner also ensures privacy and security that is consistent with applicable federal laws. In addition, the system keeps track of logins, right from the system boot, or as soon as the user profile is created.

Note When you reload a router, login notifications get reset.

Enable or disable the login banner using these commands:

Example:

```
Router(config-un)#login-history enable
Router(config-un)#login-history disable
```

Run the `show running-config username user1` command to verify the state of login banner.

```
Router(config-un)# show running-config username NAME1
Fri Jan 29 13:55:28.261 UTC
username NAME1
 group UG1
 secret * **********
 password * ******
 login-history enable
```

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Create a User Profile in System Admin VM

Create new users for the System Admin VM. Users are included in a user group and assigned certain privileges. The users have restricted access to the commands and configurations in the System Admin VM console, based on assigned privileges.

The router supports a maximum of 1024 user profiles.

The root-lr user of XR VM can access the System Admin VM by entering **Admin** command in the EXEC mode. The router does not prompt you to enter any username and password. The XR VM root-lr user is provided full access to the System Admin VM.

Step 1 **admin**

Example:

```
RP/0/RSP0/CPU0:router# admin
```

Enters administration EXEC mode.

Step 2 **config**

Example:

```
sysadmin-vm:0_RP0#config
```

Enters mode.

Step 3 **aaa authentication users user** *user_name*

Example:

```
sysadmin-vm:0_RP0(config)#aaa authentication users user us1
```

Creates a new user and enters user configuration mode. In the example, the user "us1" is created.

Step 4 **password** *password*

Example:

```
sysadmin-vm:0_RP0(config-user-us1)#password pwd1
```

Enter the password that will be used for user authentication at the time of login into System Admin VM.

Step 5 **uid** *user_id_value*

Example:

```
sysadmin-vm:0_RP0(config-user-us1)#uid 100
```

Specify a numeric value. You can enter any 32 bit integer.

Step 6 **gid** *group_id_value*

Example:

```
sysadmin-vm:0_RP0(config-user-us1)#gid 50
```

Specify a numeric value. You can enter any 32 bit integer.

Step 7 **ssh_keydir** *ssh_keydir*

Example:

```
sysadmin-vm:0_RP0(config-user-us1)#ssh_keydir dir1
```

Specify any alphanumeric value.

Step 8 **homedir** *homedir*

Example:

```
sysadmin-vm:0_RP0(config-user-us1)#homedir dir2
```

Specify any alphanumeric value.

Step 9 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Create Command Rules

Command rules are rules based on which users of a user group are either permitted or denied the use of certain commands. Command rules are associated to a user group and get applied to all users who are part of the user group.

A command rule is created by specifying whether an operation is permitted, or denied, on a command. This table lists possible operation and permission combinations:

Operation	**Accept Permission**	**Reject Permission**
Read (R)	Command is displayed on the CLI when "?" is used.	Command is not displayed on the CLI when "?" is used.
Execute (X)	Command can be executed from the CLI.	Command cannot be executed from the CLI.
Read and execute (RX)	Command is visible on the CLI and can be executed.	Command is neither visible nor executable from the CLI.

By default, all permissions are set to **Reject**.

Each command rule is identified by a number associated with it. When multiple command rules are applied to a user group, the command rule with a lower number takes precedence. For example, cmdrule 5 permits read access, while cmdrule10 rejects read access. When both these command rules are applied to the same user group, the user in this group gets read access because cmdrule 5 takes precedence.

As an example, in this task, the command rule is created to deny read and execute permissions for the "show platform" command.

Before you begin

Create an user group. See Create a User Group in System Admin VM, on page 30.

SUMMARY STEPS

1. **admin**
2. **config**
3. **aaa authorization cmdrules cmdrule** *command_rule_number*
4. **command** *command_name*
5. **ops {r | x | rx}**
6. **action** {**accept** | **accept_log** | **reject**}
7. **group** *user_group_name*

8. **context** *connection_type*
9. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **admin**

Example:

```
RP/0/RSP0/CPU0:router# admin
```

Enters administration EXEC mode.

Step 2 **config**

Example:

```
sysadmin-vm:0_RP0#config
```

Enters mode.

Step 3 **aaa authorization cmdrules cmdrule** *command_rule_number*

Example:

```
sysadmin-vm:0_RP0(config)#aaa authorization cmdrules cmdrule 1100
```

Specify a numeric value as the command rule number. You can enter a 32 bit integer.

Important Do no use numbers between 1 to 1000 because they are reserved by Cisco.

This command creates a new command rule (if it is not already present) and enters the command rule configuration mode. In the example, command rule "1100" is created.

Note By default "cmdrule 1" is created by the system when the root-system user is created. This command rule provides "accept" permission to "read" and "execute" operations for all commands. Therefore, the root user has no restrictions imposed on it, unless "cmdrule 1" is modified.

Step 4 **command** *command_name*

Example:

```
sysadmin-vm:0_RP0(config-cmdrule-1100)#command "show platform"
```

Specify the command for which permission is to be controlled.

If you enter an asterisk '*****' for **command**, it indicates that the command rule is applicable to all commands.

Step 5 **ops {r | x | rx}**

Example:

```
sysadmin-vm:0_RP0(config-cmdrule-1100)#ops rx
```

Specify the operation for which permission has to be specified:

- **r** — Read
- **x** — Execute
- **rx** — Read and execute

Step 6 **action** {**accept** | **accept_log** | **reject**}

Example:

```
sysadmin-vm:0_RP0(config-cmdrule-1100)#action reject
```

Specify whether users are permitted or denied the use of the operation.

- **accept** — users are permitted to perform the operation
- **accept_log**— users are permitted to perform the operation and every access attempt is logged.
- **reject**— users are restricted from performing the operation.

Step 7 **group** *user_group_name*

Example:

```
sysadmin-vm:0_RP0(config-cmdrule-1100)#group gr1
```

Specify the user group on which the command rule is applied.

Step 8 **context** *connection_type*

Example:

```
sysadmin-vm:0_RP0(config-cmdrule-1100)#context *
```

Specify the type of connection to which this rule applies. The connection type can be *netconf* (Network Configuration Protocol), *cli* (Command Line Interface), or *xml* (Extensible Markup Language). It is recommended that you enter an asterisk '*'; this indicates that the command rule applies to all connection types.

Step 9 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

What to do next

Create data rules. See Create Data Rules, on page 38.

Create Data Rules

Data rules are rules based on which users of the user group are either permitted, or denied, accessing and modifying configuration data elements. The data rules are associated to a user group. The data rules get applied to all users who are part of the user group.

Each data rule is identified by a number associated to it. When multiple data rules are applied to a user group, the data rule with a lower number takes precedence.

Before you begin

Create an user group. See Create a User Group in System Admin VM, on page 30.

SUMMARY STEPS

1. **admin**
2. **config**
3. **aaa authorization datarules datarule** *data_rule_number*
4. **keypath** *keypath*
5. **ops** *operation*
6. **action** {**accept** | **accept_log** | **reject**}
7. **group** *user_group_name*
8. **context** *connection type*
9. **namespace** *namespace*
10. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **admin**

Example:

```
RP/0/RSP0/CPU0:router# admin
```

Enters administration EXEC mode.

Step 2 **config**

Example:

```
sysadmin-vm:0_RP0#config
```

Enters mode.

Step 3 **aaa authorization datarules datarule** *data_rule_number*

Example:

```
sysadmin-vm:0_RP0(config)#aaa authorization datarules datarule 1100
```

Specify a numeric value as the data rule number. You can enter a 32 bit integer.

Important Do no use numbers between 1 to 1000 because they are reserved by Cisco.

This command creates a new data rule (if it is not already present) and enters the data rule configuration mode. In the example, data rule "1100" is created.

Note By default "datarule 1" is created by the system when the root-system user is created. This data rule provides "accept" permission to "read", "write", and "execute" operations for all configuration data. Therefore, the root user has no restrictions imposed on it, unless "datarule 1" is modified.

Step 4 **keypath** *keypath*

Example:

```
sysadmin-vm:0_RP0(config-datarule-1100)#keypath  /aaa/disaster-recovery
```

Specify the keypath of the data element. The keypath is an expression defining the location of the data element. If you enter an asterisk '*' for **keypath** , it indicates that the command rule is applicable to all configuration data.

Step 5 **ops** *operation*

Example:

```
sysadmin-vm:0_RP0(config-datarule-1100)#ops rw
```

Specify the operation for which permission has to be specified. Various operations are identified by these letters:

- c—Create
- d—Delete
- u—Update
- w— Write (a combination of create, update, and delete)
- r—Read
- x—Execute

Step 6 **action** {**accept** | **accept_log** | **reject**}

Example:

```
sysadmin-vm:0_RP0(config-datarule-1100)#action reject
```

Specify whether users are permitted or denied the operation.

- **accept** — users are permitted to perform the operation
- **accept_log**— users are permitted to perform the operation and every access attempt is logged
- **reject**— users are restricted from performing the operation

Step 7 **group** *user_group_name*

Example:

```
sysadmin-vm:0_RP0(config-datarule-1100)#group gr1
```

Specify the user group on which the data rule is applied. Multiple group names can also be specified.

Step 8 **context** *connection type*

Example:

```
sysadmin-vm:0_RP0(config-datarule-1100)#context *
```

Specify the type of connection to which this rule applies. The connection type can be *netconf* (Network Configuration Protocol), *cli* (Command Line Interface), or *xml* (Extensible Markup Language). It is recommended that you enter an asterisk '*', which indicates that the command applies to all connection types.

Step 9 **namespace** *namespace*

Example:

```
sysadmin-vm:0_RP0(config-datarule-1100)#namespace *
```

Enter asterisk '*' to indicate that the data rule is applicable for all namespace values.

Step 10 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Change Disaster-recovery Username and Password

When you define the root-system username and password initially after starting the router, the same username and password gets mapped as the disaster-recovery username and password for the System Admin console. However, it can be changed.

The disaster-recovery username and password is useful in these scenarios:

- Access the system when the AAA database, which is the default source for authentication in System Admin console is corrupted.
- Access the system through the management port, when, for some reason, the System Admin console is not working.
- Create new users by accessing the System Admin console using the disaster-recovery username and password, when the regular username and password is forgotten.

Note On the router, you can configure only one disaster-recovery username and password at a time.

SUMMARY STEPS

1. **admin**
2. **config**
3. **aaa disaster-recovery username** *username* **password** *password*
4. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **admin**

Example:

```
RP/0/RSP0/CPU0:router# admin
```

Enters administration EXEC mode.

Step 2 **config**

Example:

```
sysadmin-vm:0_RP0#config
```

Enters mode.

Step 3 **aaa disaster-recovery username** *username* **password** *password*

Example:

```
sysadmin-vm:0_RP0(config)#aaa disaster-recovery username us1 password pwd1
```

Specify the disaster-recovery username and the password. You have to select an existing user as the disaster-recovery user. In the example, 'us1' is selected as the disaster-recovery user and assigned the password as 'pwd1'. The password can be entered as a plain text or md5 digest string.

When you need to make use of the disaster recovery username, you need to enter it as *username***@localhost**.

Step 4 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Recover Password using PXE Boot

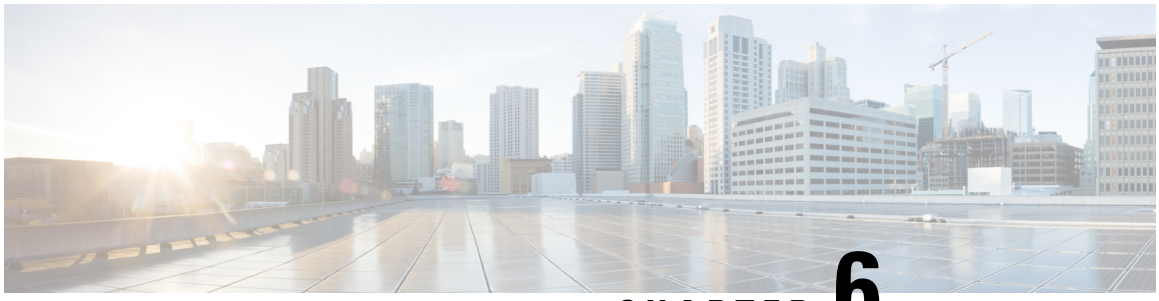
If you are unable to login or lost your XR and System administration passwords, use the following steps to create new password. A lost password cannot be recovered, instead a new username and password must be created with a non-graceful PXE boot.

Step 1 Boot the router using PXE.

Note PXE boot is fully intrusive. The router state, configuration and image is reset.

To PXE boot a router, see Boot the Router Using iPXE, on page 10.

Step 2 Reset the password.



CHAPTER 6

Perform System Upgrade and Install Feature Packages

The system upgrade and package installation processes are executed using **install** commands on the router. The processes involve adding and activating the iso images (*.iso*) and feature packages on the router. These files are accessed from a network server and then activated on the router. If the installed package or SMU causes any issue on the router, it can be uninstalled.

The topics covered in this chapter are:

Upgrading the System

Upgrading the system is the process of installing a new version of the Cisco IOS XR operating system on the router. The router comes preinstalled with the Cisco IOS XR image. However, you can install the new version in order to keep router features up to date. The system upgrade operation is performed from the XR VM. However, during system upgrade, the software that runs on both the XR VM and the System Admin VM get upgraded.

Note If an interface on a router doesn't have a configuration and is brought up by performing no-shut operation, then upon router reload, the interface state changes to **admin-shutdown** automatically.

Note

- Ensure that you have adequate disk space.
- Run the **fsck** command to check the status of the file system, for a successful IOS XR upgrade. You must run the **fsck** command in the System Admin EXEC mode to install a System Admin package, and in the XR EXEC mode to install the XR package.
- All install commands are applicable in both the System Admin EXEC mode and in XR EXEC mode. System Admin install operations are done from XR EXEC mode.

Perform a system upgrade by installing a base package–Cisco IOS XR Unicast Routing Core Bundle. To install this bundle, run the **install** command. The filename for the Cisco IOS XR Unicast Routing Core Bundle bundle is *asr9k-mini-x.iso*.

Caution Do not perform any install operations when the router is reloading.

Do not reload the router during an upgrade operation.

Note CSM Server is a web-based, server-side automation and orchestration framework. It gives service providers the ability to simultaneously schedule and deploy SMUs and perform software upgrades across hundreds of routers in a scheduled manner through a simple click Web interface. For more information, see Cisco Software Manager.

Note If you perform a manual or automatic system reload without completing the transaction with the **install commit** command, the action will revert the system to the point before the install transaction commenced, including any configuration changes. Only the log is preserved for debugging.

This action clears all configuration rollback points available. You'll not be able to roll back to, or view, any commits made until the install rollback event. Any new commits made after the install rollback event starts from commit ID '1000000001'.

Note To enable hardware programming after upgrading the chassis from an older software version to IOS XR Release 7.6.x or later through ISSU, initiate a chassis reload. The chassis reload is mandatory, if you must enable a maximum transmission unit (MTU) value of 9646 on applicable interfaces.

View Supported Software Upgrade or Downgrade Versions

Table 2: Feature History Table

Feature Name	Release Information	Description
Supported Software Upgrade or Downgrade IOS XR Versions	Release 7.5.1	You can determine whether a software version can be upgraded or downgraded to another version using this functionality. Before an actual upgrade or downgrade process, you can also view the hardware or software limitations that could cause the upgrade or downgrade to fail. This feature helps you plan successful software upgrades or downgrades. This feature introduces the **show install upgrade-matrix** command.

Your Cisco router comes preinstalled with IOS XR software. You either upgrade the software release to use new features and software fixes, or you downgrade the software. To leverage new features that are added or software fixes that are provided, it is important that you upgrade your router to a current version.

To help you select a Cisco IOS XR software release that aligns with Cisco-certified upgrade and downgrade paths, this feature provides answers to the following questions:

- What upgrade or downgrade releases are supported for the current release?
- I plan to upgrade from Release X to Release Y. Does my router support upgrade to Release Y?
- Are there any bridging SMUs that must be installed before I upgrade the software?

This feature provides a mechanism to determine whether the current release supports an upgrade to a target release. This task is run at the start of a software upgrade or downgrade through the **install replace** command. If the validation fails, the software upgrade is blocked, and the system notifies the reason for the failure. This feature allows you to proactively examine whether you can upgrade or downgrade to a certain release, saving time and effort involved in planning and upgrading the software.

The feature provides the following information to help you understand the prerequisites or limitations related to the specific software upgrade or downgrade:

- Required bridging SMU RPMs
- Blocking SMU RPMs
- Unsupported hardware
- Caveats or restrictions

You can overwrite the automatic validation using the **force** keyword in the **install replace** command. With this option, the system displays warning messages when the upgrade fails but does not block the software

upgrade. Use the **force ?** keyword to understand any other impact to system functionalities apart from the disabling of this process that determines the supported releases for software upgrade or downgrade.

You can view the support information using the following **show** commands or through the operational data.

Command	Description
show install upgrade-matrix running	Displays all supported software upgrades from the current version according to the support data installed on the running system
show install upgrade-matrix iso *path-to-ISO*	Displays details about the software upgrade from the current version to the version of the target ISO according to the support data in both the running system and the ISO image
show install upgrade-matrix iso *path-to-ISO* **all**	Displays all supported software upgrades from any version according to the support data in the target ISO image
show install upgrade-matrix iso *path-to-ISO* **from-running**	Displays details about the software upgrade from the current version to the version of ISO according to the support matrices in both the running system and the target ISO image

View All Supported Software Upgrade from Running Version

The following example shows all supported releases for upgrade from the current version 7.5.1 on the router:

```
Router#show install upgrade-matrix running
Fri Jul 29 10:18:43.413 IST
This may take a while ...

The current software [7.5.1] can be upgraded from and downgraded to the following releases:

================================================================================
From      To        Bridge SMUs Required        Caveats
================================================================================
7.0.2     7.5.1     r702.CSCvw57276             None
--------------------------------------------------------------------------------
6.5.3     7.5.1     r653.CSCvw57276             None
--------------------------------------------------------------------------------
7.5.1     7.0.2     None                        None
--------------------------------------------------------------------------------
7.5.1     6.5.3     None                        - https://www.cisco.com/c/en/us/
                                                td/docs/routers/asr9000/software
                                                /asr9k-r7-4/system-
                                                security/configuration/guide/b
                                                -system-security-cg-asr9000-74x
                                                /configuring-aaa-
                                                services.html#id_128191
--------------------------------------------------------------------------------
7.5.1     7.4.1     None                        None
--------------------------------------------------------------------------------
7.5.1     7.1.25    None                        None
--------------------------------------------------------------------------------
```

```
7.5.1     7.1.3     None                          None
--------------------------------------------------------------------------------
7.5.1     7.1.2     None                          None
--------------------------------------------------------------------------------
7.5.1     7.3.1     None                          None
--------------------------------------------------------------------------------
7.5.1     6.6.3     None                          - https://www.cisco.com/c/en/us/
                                                  td/docs/routers/asr9000/software
                                                  /asr9k-r7-4/system-
                                                  security/configuration/guide/b
                                                  -system-security-cg-asr9000-74x
                                                  /configuring-aaa-
                                                  services.html#id_128191
--------------------------------------------------------------------------------
7.5.1     7.3.2     None                          None
--------------------------------------------------------------------------------
7.4.1     7.5.1     None                          None
--------------------------------------------------------------------------------
7.1.25    7.5.1     None                          None
--------------------------------------------------------------------------------
7.1.3     7.5.1     None                          None
--------------------------------------------------------------------------------
7.1.2     7.5.1     None                          None
--------------------------------------------------------------------------------
7.3.1     7.5.1     None                          None
--------------------------------------------------------------------------------
6.6.3     7.5.1     r663.CSCvw57276               None
--------------------------------------------------------------------------------
7.3.2     7.5.1     None                          None
--------------------------------------------------------------------------------
```

View Supported Releases to Upgrade Software From Current Version to Target Version

This example shows the supported release to upgrade software from the current version to a target version.

```
Router#show install upgrade-matrix iso /harddisk:/asr9k-goldenk9-x64-7.5.2-rev1.iso

Fri Jul 29 10:19:24.185 IST
This may take a while ...
Upgrade from the current software [7.5.1] to 7.5.2 is supported
================================================================================
From      To        Bridge SMUs Required        Caveats
================================================================================
7.5.1     7.5.2     None                        None
--------------------------------------------------------------------------------
```

The current image has the upgrade matrix that specifies only its supported upgrade or downgrade versions up to a certain version. If you want to determine the upgrade path of a newer version of ISO that is higher than the version in the current matrix, the upgrade matrix from the new ISO provides the supported upgrade or downgrade paths.

View Supported Releases from Current Version to an ISO Version

The following example shows the software upgrade paths, downgrade paths, and restrictions to an upgrade from the current version to the target ISO version:

```
Router#show install upgrade-matrix iso /harddisk:/ asr9k-goldenk9-x64-7.5.2-rev1.iso all
Fri Jul 29 10:20:10.961 IST
This may take a while ...
```

7.5.2 can be upgraded from and downgraded to the following releases:

From	To	Bridge SMUs Required	Caveats
7.0.2	7.5.2	r702.CSCvw57276	None
7.5.1	7.5.2	None	None
7.4.2	7.5.2	None	None
7.4.1	7.5.2	None	None
7.5.2	7.0.2	None	None
7.5.2	6.5.3	None	- https://www.cisco.com/c/en/us/td/docs/routers/asr9000/software/asr9k-r7-4/system-security/configuration/guide/b-system-security-cg-asr9000-74x/configuring-aaa-services.html#id_128191
7.5.2	7.1.25	None	None
7.5.2	7.4.2	None	None
7.5.2	7.6.1	None	None
7.5.2	7.4.1	None	None
7.5.2	7.1.3	None	None
7.5.2	7.1.2	None	None
7.5.2	7.3.1	None	None
7.5.2	6.6.3	None	- https://www.cisco.com/c/en/us/td/docs/routers/asr9000/software/asr9k-r7-4/system-security/configuration/guide/b-system-security-cg-asr9000-74x/configuring-aaa-services.html#id_128191
7.5.2	7.3.2	None	None
7.1.25	7.5.2	None	None
7.1.3	7.5.2	None	None
7.1.2	7.5.2	None	None
7.6.1	7.5.2	None	None
6.5.3	7.5.2	r653.CSCvw57276	None
7.3.1	7.5.2	None	None
6.6.3	7.5.2	r663.CSCvw57276	None
7.3.2	7.5.2	None	None

View Supported Releases from Running Version to an ISO Version

The following example displays details about the software upgrade from the current version to the version of ISO according to the support matrices in both the running system and the target ISO image:

```
Router#show install upgrade-matrix iso /harddisk:/asr9k-goldenk9-x64-7.5.2-rev1.iso
from-running
Fri Jul 29 10:21:31.957 IST
This may take a while ...
Upgrade from the current software [7.5.1] to 7.5.2 is supported
================================================================================
From      To        Bridge SMUs Required        Caveats
================================================================================
7.5.1     7.5.2     None                        None
--------------------------------------------------------------------------------
```

Upgrading Features

Upgrading features is the process of deploying new features and software patches on the router. Perform a feature upgrade by installing packages. Perform a software patch installation by installing Software Maintenance Upgrade (SMU) files.

Installing a package on the router installs specific features that are part of that package. Cisco IOS XR Software is divided into various software packages; this enables you to select the features to run on your router. Each package contains components that perform a specific set of router functions, such as routing, security, and so on.

For example, the components of the routing package are split into individual RPMs such as BGP and OSPF. BGP is a part of the base software version and is a mandatory RPM, and hence can't be removed. However, you can add and remove optional RPMs such as OSPF as required.

The naming convention of the package is `<platform>-<pkg>-<pkg version>-<release version>.<architecture>.rpm`.

For example:

```
asr9k-9000v-nV-x64-1.0.0.0-r702.x86_64.rpm
```

Use the **install** commands to install packages and SMUs. For more information about the install process, see Install Packages, on page 54.

Note

- Ensure that you have adequate disk space.
- Run the **fsck** command to check the status of the file system, for a successful IOS XR upgrade. You must run the **fsck** command in the System Admin EXEC mode to install a System Admin package, and in the XR EXEC mode to install the XR package.
- All install commands are applicable in both the System Admin EXEC mode and in XR EXEC mode. System Admin install operations are done from XR EXEC mode.

There are separate packages and SMUs for the XR VM and the System Admin VM. They can be identified by their filenames.

For example, `asr9k-px-7.9.1.CSCvu59908.pie` is an example of a package for the XR VM. `asr9k-sysadmin-7.9.1.pie` is associated with the system admin VM.

The XR packages or SMUs are activated from the XR VM, whereas the System Admin packages or SMUs are activated from the System Admin VM.

You can alternatively perform a cross VM operation, by activating or deactivating the System Admin packages and SMUs from XR.

Optimized Size of Install Image

Table 3: Feature History Table

Feature Name	Release Information	Description
Optimized Size of Install Image	Release 7.3.1	With this release, the size of the Cisco ISO image installed on the router is reduced by approximately 300 MB. This frees-up the disk space on the router, and speeds-up time taken for installing the package.

Cisco IOS XR, Release 7.3.1 introduces an optimized version of mini-x ISO image with the overall image size reduced by approximately 300MB. This optimized ISO reduces the time for install operations, and the utilization of disk space for files stored in the install repository.

This optimized ISO supports booting the router using the following methods:

- iPXE boot
- USB boot
- System upgrade
- In-Service Software Upgrade (ISSU)

For ISSU, upgrading from a lower version to release 7.3.1 or later requires a bridge SMU in the running lower version. Whereas rolling back from release 7.3.1 or later to a lower version does not require a bridge SMU. There are no SMU dependencies when downgrading from release 7.3.1 with SMU installed to an older version. The bridge SMU is a mandatory prerequisite and must be installed on the running version before an upgrade is performed to release 7.3.1 or later.

The following are the bridge SMUs for host, XR and System admin that must be installed on the lower running version. For example, here, release 7.1.1:

- asr9k-iosxr-infra-64-3.0.0.2-r711.CSCvq46421.x86_64.rpm
- asr9k-iosxr-os-64-3.0.0.2-r711.CSCvq46421.x86_64.rpm
- asr9k-sysadmin-system-7.1.1-r711.CSCvq46421.x86_64.rpm

Bridge SMUs are included in the Cisco IOS XR software tar bundles located in Download Software Center for the specific release.

The commands related to install operations will display a new package with the term `common`. The common package is also listed in the output of install commands when executed using NETCONF Yang mode.

Note The `common` term is not listed for **show install active summary** and **show install committed summary** commands.

The common package is available in `/misc/disk1/tftpboot` of System admin VM.

The term `common` is also displayed in the output of **show install repository** or **show install repository all** commands in Sysadmin VM and XR VM.

The following example shows the output from Sysadmin VM:

```
sysadmin-vm:0_RSP0# show install repository
Sat May 2 16:33:25.200 UTC+00:00
Admin repository
---------------------
asr9k-common-7.3.1
asr9k-goldenk9-x64-7.3.1-supersetOptInstallSmu
asr9k-mini-x64-7.3.1
asr9k-sysadmin-7.3.1
asr9k-sysadmin-hostos-7.3.1-r731.admin.x86_64
asr9k-sysadmin-hostos-7.3.1-r731.host.x86_64
asr9k-sysadmin-hostos-7.3.1-r731.CSCho77777.admin.x86_64
asr9k-sysadmin-hostos-7.3.1-r731.CSCho77777.host.x86_64
asr9k-sysadmin-hostos-7.3.1-r731.admin.x86_64
asr9k-sysadmin-hostos-7.3.1-r731.host.x86_64
asr9k-sysadmin-hostos-7.3.1-r731.CSCho77777.admin.x86_64
asr9k-sysadmin-hostos-7.3.1-r731.CSCho77777.host.x86_64
asr9k-sysadmin-shared-7.3.1-r731.CSCcv33333.x86_64
asr9k-sysadmin-shared-7.3.1-r731.CSCcv33333.x86_64
asr9k-sysadmin-system-7.3.1-r731.CSCcv11111.x86_64
asr9k-sysadmin-system-7.3.1-r731.CSCcv22222.x86_64
---------------------- Truncated for Brevity -------------------------
```

The following example shows the output from XR VM:

```
Router#show install repository
Sat May 2 13:55:59.996 UTC
26 package(s) in XR repository:
asr9k-iosxr-infra-64-1.0.0.1-r731.CSCxr22222.x86_64
asr9k-iosxr-infra-64-1.0.0.2-r731.CSCxr44444.x86_64
asr9k-k9sec-x64-2.1.0.0-r731.x86_64
asr9k-optic-x64-1.0.0.0-r731.x86_64
asr9k-mgbl-x64-2.0.0.0-r731.x86_64
asr9k-mpls-te-rsvp-x64-2.1.0.0-r731.x86_64
asr9k-common-7.3.1
asr9k-goldenk9-x64-7.3.1-supersetOptInstallSmu
asr9k-bng-x64-1.0.0.0-r731.x86_64
asr9k-mini-x64-7.3.1
asr9k-mpls-x64-2.0.0.0-r731.x86_64
asr9k-li-x64-1.1.0.0-r73104I.x86_64
---------------------- Truncated for Brevity -------------------------
```

In the command output, `asr9k-common-7.3.1` represents the optimized package to reduce the image size.

Install Prepared Packages

A system upgrade or feature upgrade is performed by activating the ISO image file, packages, and SMUs. It is possible to prepare these installable files before activation. During the prepare phase, preactivation checks are made and the components of the installable files are loaded on to the router setup. The prepare process runs in the background and the router is fully usable during this time. When the prepare phase is over, all the prepared files can be activated instantaneously. The advantages of preparing before activation are:

- If the installable file is corrupted, the prepare process fails. This provides an early warning of the problem. If the corrupted file was activated directly, it might cause router malfunction.
- Directly activating an ISO image for system upgrade takes considerable time during which the router is not usable. However, if the image is prepared before activation, not only does the prepare process run asynchronously, but when the prepared image is subsequently activated, the activation process too takes less time. As a result, the router downtime is considerably reduced.
- It performs a disk-space check that is required for a successful operation. This quantifies the disk-space deficit, and provides you possible alternatives to free up space in the filesystem.
- It performs a package compatibility check. This ensures that all the required installation packages are available. For any package compatibility check error, details of the package and version are logged.

Complete this task to upgrade the system and install packages by making use of the prepare operation.

Note Depending on whether you are installing a System Admin package or a XR package, execute the **install** commands in the mode or mode respectively. All **install** commands are applicable in both these modes. System Admin install operations can be done from XR mode.

Step 1 Add the required ISO image and packages to the repository.

For details, see Install Packages, on page 54.

Step 2 **show install repository**

Example:

```
RP/0/RSP0/CPU0:router#show install repository
```

Perform this step to verify that the required installable files are available in the repository. Packages are displayed only after the "install add" operation is complete.

Step 3 **show install request**

Example:

```
RP/0/RSP0/CPU0:router#show install request
```

(Optional) Displays the operation ID of the add operation and its status. The operation ID can be later used to execute the **activate** command.

```
Install operation 8 is still in progress
```

Step 4 Execute one of these:

- **install prepare** *package_name*

- **install prepare id** *operation_id*

Example:

The prepare process takes place. This operation is performed in asynchronous mode. The **install prepare** command runs in the background, and the EXEC prompt is returned as soon as possible.

If you use the operation ID, all packages that were added in the specified operation are prepared together. For example, if 5 packages are added in operation 8, by executing **install prepare id 8**, all 5 packages are prepared together. You do not have to prepare the packages individually.

Step 5 **show install prepare**

Example:

```
RP/0/RSP0/CPU0:router#show install prepare
```

Displays packages that are prepared. From the result, verify that all the required packages have been prepared.

Step 6 **install activate**

Example:

```
RP/0/RSP0/CPU0:router#install activate
```

All the packages that have been prepared are activated together to make the package configurations active on the router.

Note You should not specify any package name or operation ID in the CLI.

Activations of some SMUs require manual reload of the router. When such SMUs are activated, a warning message is displayed to perform reload. The components of the SMU get activated only after the reload is complete. Perform router reload immediately after the execution of the **install activate** command is completed.

Step 7 **show install active**

Example:

```
RP/0/RSP0/CPU0:router#show install active
```

Displays packages that are active.

From the result, verify that on all RPs and LCs, the same image and package versions are active.

Step 8 **install commit**

Example:

```
RP/0/RSP0/CPU0:router#install commit
```

Installing Packages: Related Commands

Related Commands	Purpose
show install log	Displays the log information for the install process; this can be used for troubleshooting in case of install failure.
show install package	Displays the details of the packages that have been added to the repository. Use this command to identify individual components of a package.
install prepare clean	Clears the prepare operation and removes all the packages from the prepared state.

What to do next

- After performing a system upgrade, upgrade FPD by using the **upgrade hw-module location all fpd all** command from the mode. The progress of FPD upgrade process can be monitored using the **show hw-module fpd** command in the mode. Reload the router after the FPD upgrade is completed.
- Verify the installation using the **install verify packages** command.
- Uninstall the packages or SMUs if their installation causes any issues on the router. See Uninstall Packages.

Note ISO images cannot be uninstalled. However, you can perform a system downgrade by installing an older ISO version.

Install Packages

Complete this task to upgrade the system or install a patch. The system upgrade is done using an ISO image file, while the patch installation is done using packages and SMUs. You can also include SMUs in an upgrade operation along with mini ISO.

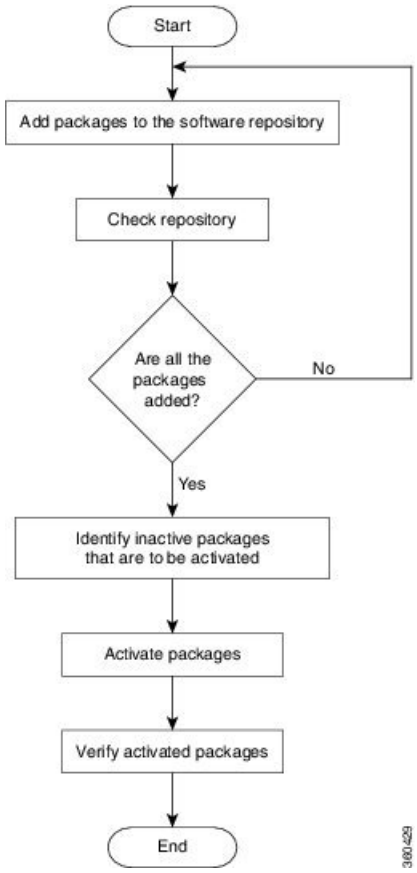
This task is also used to install *.rpm* files. The *.rpm* file contains multiple packages and SMUs that are merged into a single file. The packaging format defines one RPM per component, without dependency on the card type.

Note
- Ensure that you have adequate disk space.
- Run the **fsck** command to check the status of the file system, for a successful IOS XR upgrade. You must run the **fsck** command in the System Admin EXEC mode to install a System Admin package, and in the XR EXEC mode to install the XR package.
- All install commands are applicable in both the System Admin EXEC mode and in XR EXEC mode. System Admin install operations are done from XR EXEC mode.

Note
- The system upgrade is supported only from XR EXEC mode.
- While the System Admin package can be executed using **install** commands in the System Admin EXEC mode and XR EXEC mode, the XR package can only be executed using the install commands in XR EXEC mode. All **install** commands are applicable in both these modes.
- While the System Admin SMUs can be installed in System Admin EXEC mode and XR EXEC mode, the XR SMUs can only be installed through the XR EXEC mode.
- Install operation over IPv6 is not supported.

The workflow for installing a package is shown in this flowchart.

Figure 3: Installing Packages Workflow



Before you begin

- Configure and connect to the management port. The installable file is accessed through the management port.
- Copy the package to be installed either on the router's hard disk or on a network server to which the router has access.

Step 1 Execute one of these:

- **install add source** *<http or shttp transfer protocol>/package_path/ filename1 filename2 ...*
- **install add source** *<tftp transfer protocol>/package_path/ filename1 filename2 ...*
- **install add source** *<ftp or sftp transfer protocol>//user@server:/package_path/ filename1 filename2 ...*

Example:

```
RP/0/RSP0/CPU0:router#install add source /harddisk:/ asr9k-mpls-x64-2.1.0.0-r731.x86_64.rpm
asr9k-mpls-x64-2.1.0.0-r732.x86_64.rpm
```

or

```
RP/0/RSP0/CPU0:router#install add source sftp://root@8.33.5.15:/auto/ncs/package/
asr9k-mcast-1.0.0.0-731.x86_64.rpm asr9k-iosxr-mpls-1.0.0.0-732.x86_64.rpm
```

or

```
RP/0/RSP0/CPU0:router#install add source /harddisk:/ asr9k-mpls-x64-2.1.0.0-<release-number>.x86_64.rpm
 asr9k-mpls-x64-2.1.0.0-<release-number>.x86_64.rpm
```

or

```
RP/0/RSP0/CPU0:router#install add source sftp://root@8.33.5.15:/auto/ncs/package/
asr9k-mcast-1.0.0.0-<release-number>.x86_64.rpm asr9k-iosxr-mpls-1.0.0.0-<release-number>.x86_64.rpm
```

Note A space must be provided between the *package_path* and *filename*.

The software files are unpacked from the package, validated, and then added to the software repository. This operation might take time depending on the size of the files being added. The operation is performed in asynchronous mode. The **install add** command runs in the background, and the EXEC prompt is returned when all files are unpacked.

Note The repositories for the XR VM and the System Admin VM are different. The system automatically adds a routing package to the XR VM repository and a system administration package to the System Admin VM repository.

Step 2 **show install request**

Example:

```
RP/0/RSP0/CPU0:router#show install request
```

(Optional) Displays the operation ID of the add operation and its status. The operation ID can be later used to execute the **activate** command.

```
Install operation 8 is still in progress
```

Step 3 **show install repository**

Example:

```
RP/0/RSP0/CPU0:router#show install repository
```

Displays packages that are added to the repository. Packages are displayed only after the `install add` operation is complete.

Step 4 **show install inactive**

Example:

```
RP/0/RSP0/CPU0:router#show install inactive
```

Displays inactive packages that are present in the repository. Only inactive packages can be activated.

Step 5 Execute one of these:

- **install activate** *package_name*
- **install activate id** *operation_id*

Example:

The *operation_id* is that of the **install add** operation, see Install Packages, on page 54 Step Step 2, on page 56. This command can also be run from the Sys Admin mode. The package configurations are made active on the router. As a result, new features and software fixes take effect. This operation is performed in asynchronous mode, as this is the default. The **install activate** command runs in the background, and the EXEC prompt is returned.

You can run the activate operation either through the synchronous mode or by selecting the `sync` option from the CLI.

If you use the operation ID, all packages that were added in the specified operation are activated together. For example, if 5 packages are added in operation ID 8, by executing **install activate id 8**, all 5 packages are activated together. You do not have to activate the packages individually.

Activation does not happen instantaneously, but takes some time. When activation completes, the system reloads automatically. For restart SMU activation, the SMU takes effect once the processes impacted by the SMU are restarted.

If the SMU has dependency on both XR VM and System Admin VM, perform the reload after activating the SMU in both VMs so that they take effect simultaneously. To reload the router, use the **hw-module location all reload** command from the System Admin EXEC mode.

Step 6 **show install active**

Example:

```
RP/0/RSP0/CPU0:router#show install active
```

Displays packages that are active.

From the result, verify that the same image and package versions are active on all RPs and LCs.

Table 4: Example: Installing Packages: Related Commands

Related Commands	Purpose
show install log	Displays the log information for the install process; this can be used for troubleshooting in case of install failure.
show install package	Displays the details of the packages that have been added to the repository. Use this command to identify individual components of a package.
install prepare	Makes pre-activation checks on an inactive package, to prepare it for activation.
show install prepare	Displays the list of package that have been prepared and are ready for activation.

Step 7 **install commit**

Example:

```
RP/0/RSP0/CPU0:router#install commit
```

Commits the Host, XR, and System Admin newly active software.

Note On Multi-SDR mode, you can use the **install commit sdr** to commit just the sdr from where the CLI is being triggered. For more information, see Secure Domain Router Commands.

What to do next

- After performing a system upgrade, upgrade FPD by using the **upgrade hw-module location all fpd all** command from the mode. The progress of FPD upgrade process can be monitored using the **show hw-module fpd** command in the mode. Reload the router after the FPD upgrade is completed.
- Verify the installation using the **install verify packages** command.
- Uninstall the packages or SMUs if their installation causes any issues on the router. See Uninstall Packages, on page 58.

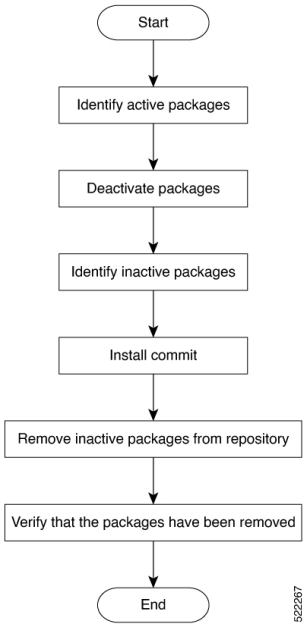
Uninstall Packages

Complete this task to uninstall a package. All router functionalities that are part of the uninstalled package are deactivated. Packages that are added in the XR VM cannot be uninstalled from the System Admin VM. However, the cross VM operation allows System Admin packages to be deactivated from XR as well.

Note Installed ISO images cannot be uninstalled. Also, kernel SMUs that install third party SMU on host, XR VM and System Admin VM, cannot be uninstalled. However, subsequent installation of ISO image or kernel SMU overwrites the existing installation.

The workflow for uninstalling a package is shown in this flowchart.

Figure 4: Uninstalling Packages Workflow



This task uninstalls XR VM packages. If you need to uninstall System Admin packages, run the same commands from the mode.

Step 1 **show install active**

Example:

```
RP/0/RSP0/CPU0:router#show install active
```

Displays active packages. Only active packages can be deactivated.

Step 2 Execute one of these:

- **install deactivate** *package_name*
- **install deactivate id** *operation_id*

Example:

The *operation_id* is the ID from **install add** operation. All features and software patches associated with the package are deactivated. You can specify multiple package names and deactivate them simultaneously.

If you use the operation ID, all packages that were added in the specified operation are deactivated together. You do not have to deactivate the packages individually. If System admin packages were added as a part of the **install add** operation (of the ID used in deactivate) then those packages will also be deactivated.

Step 3 **show install inactive**

Example:

```
RP/0/RSP0/CPU0:router#show install inactive
```

The deactivated packages are now listed as inactive packages. Only inactive packages can be removed from the repository.

Step 4 **install commit**

Step 5 **install remove** *package_name*

Example:

The inactive packages are removed from the repository.

Use the **install remove** command with the **id** *operation-id* keyword and argument to remove all packages that were added for the specified operation ID.

You can also use the **install remove inactive all** to remove all inactive packages from XR and System Admin.

Step 6 **show install repository**

Example:

```
RP/0/RSP0/CPU0:router#show install repository
```

Displays packages available in the repository. The package that are removed are no longer displayed in the result.

What to do next

Install required packages. .

View Features and Capabilities Supported on a Platform

Table 5: Feature History Table

Feature Name	Release Information	Description
View Features and Capabilities Supported on a Platform	Release 7.5.2	This functionality displays a list of supported and unsupported features and their capabilities in a release for your router. With this feature, you are better equipped to plan your network configuration with features annotated for their support information. This feature introduces the **show features** command.

This feature provides an answer to the question `Is feature X supported on my router?`

You can determine whether a feature and their capabilities are supported on your router for the release. The support information is based on the release and platform-specific data such as platform variants, RP, or LC present on the router.

Note In Cisco IOS XR Software Release 7.5.2, only the capabilities for Access Control List (ACL) feature is supported.

The functionality to determine the capabilities information is enabled by default when the supported release is installed on the router.

Use the **show features** command to view the list of supported features and their capabilities. The feature capabilities are displayed in a tree structure with notations for the support information. For example, in ACL, the capability to use compression to accommodate a large number of Access Control Elements (ACEs) is supported, whereas IPv6 ACL BNG does not have support data in Cisco IOS XR Software Release 7.5.2. This support information about the feature is represented with the following key in the tree structure:

Key	Capability Support Information	Description
X	Unsupported	The feature capability is not supported on the platform for the release
-	Supported	The feature capability is supported on the platform for the release
?	Support unknown	The support for the feature capability is unknown on the platform for the release. This data could be because the optional package for the feature is not installed on the router.
*	Support data not available	The support for the feature capability is not available on the platform for the release. This data could be because the feature may be specific to a line card that is not present on the router.

View the List of Supported Features

In this example, the supported features on the router are displayed.

Note In Cisco IOS XR Software Release 7.5.2, only the feature capabilities for Access Control List (ACL) are supported.

```
Router#show features
Fri June 3 19:16:58.298 UTC
Key:
X - Unsupported
- - Supported
? - Support unknown (optional package not installed)
* - Support data not available

[-] Cisco IOS XR
|--[-] XR Protocols
|  |--[-] XR Base Protocols
```

```
|   |   |--[-] Services
|   |   |   |--[-] Access Control List (ACL)
|   |   |   |   |--[-]  IPv6 ACL Support
|   |   |   |   |   |--[*] IPv6 ACL ABF Track
|   |   |   |   |   |--[*] IPv6 ACL BNG
|   |   |   |   |   |--[*] IPv6 ACL Chaining (Meta ACL)
|   |   |   |   |   |--[-] IPv6 ACL Common ACL
|   |   |   |   |   |--[-] IPv6 ACL Compression
|   |   |   |   |   |--[*] IPv6 ACL Default ABF
|   |   |   |   |   |--[*] IPv6 ACL Fragment
|   |   |   |   |   |--[-] IPv6 ACL ICMP Off
|   |   |   |   |   |--[-] IPv6 ACL ICMP Protocol
|   |   |   |   |   |--[-] IPv6 ACL Interface Statistics
|   |   |   |   |   |--[-] IPv6 ACL Log Rate
|   |   |   |   |   |--[-] IPv6 ACL Log Threshold
|   |   |   |   |   |--[-] IPv6 ACL Logging
|   |   |   |   |   |--[-] IPv6 ACL MIB
|   |   |   |   |   |--[-] IPv6 ACL Object Groups (Scale)
|   |   |   |   |   |--[-] IPv6 ACL Police
|   |   |   |   |   |--[-] IPv6 ACL Priority
|   |   |   |   |   |--[*] IPv6 ACL Protocol Range
|   |   |   |   |   |--[-] IPv6 ACL Set Qos-Group
|   |   |   |   |   |--[-] IPv6 ACL Set TTL
|   |   |   |   |   |--[-] IPv6 ACL TCP Flags
|   |   |   |   |   |--[-] IPv6 ACL TTL Match
|   |   |   |   |   |--[-] IPv6 ACL UDF
|   |   |   |   |--[-] ES-ACL Support (L2 ACL)
|   |   |   |   |--[-] IPv4 ACL Support
|   |   |   |   |   |--[-] IPv4 ACL  Set Qos-group
|   |   |   |   |   |--[*] IPv4 ACL ABF Track
|   |   |   |   |   |--[*] IPv4 ACL BNG
|   |   |   |   |   |--[*] IPv4 ACL Chaining (Meta ACL)
|   |   |   |   |   |--[-] IPv4 ACL Common ACL
|   |   |   |   |   |--[-] IPv4 ACL Compression
|   |   |   |   |   |--[*] IPv4 ACL Default ABF
|   |   |   |   |   |--[*] IPv4 ACL Fragment
|   |   |   |   |   |--[-] IPv4 ACL Fragment Flags
|   |   |   |   |   |--[-] IPv4 ACL ICMP Off
|   |   |   |   |   |--[-] IPv4 ACL ICMP Protocol
|   |   |   |   |   |--[-] IPv4 ACL Interface Statistics
|   |   |   |   |   |--[-] IPv4 ACL Log Rate
|   |   |   |   |   |--[-] IPv4 ACL Log Threshold
|   |   |   |   |   |--[-] IPv4 ACL Logging
|   |   |   |   |   |--[-] IPv4 ACL MIB
|   |   |   |   |   |--[-] IPv4 ACL Object Groups (Scale)
|   |   |   |   |   |--[-] IPv4 ACL Police
|   |   |   |   |   |--[-] IPv4 ACL Priority
|   |   |   |   |   |--[*] IPv4 ACL Protocol Range
|   |   |   |   |   |--[-] IPv4 ACL Set TTL
|   |   |   |   |   |--[-] IPv4 ACL TCP Flags
|   |   |   |   |   |--[-] IPv4 ACL TTL
|   |   |   |   |   |--[-] IPv4 ACL UDF
|   |   |   |   |--[-] IPv4 Prefix-List
|   |   |   |   |--[-] IPv6 Prefix-List
```

View the List of Supported ACL Features

In this example, the capabilities for ACL features on the router are displayed.

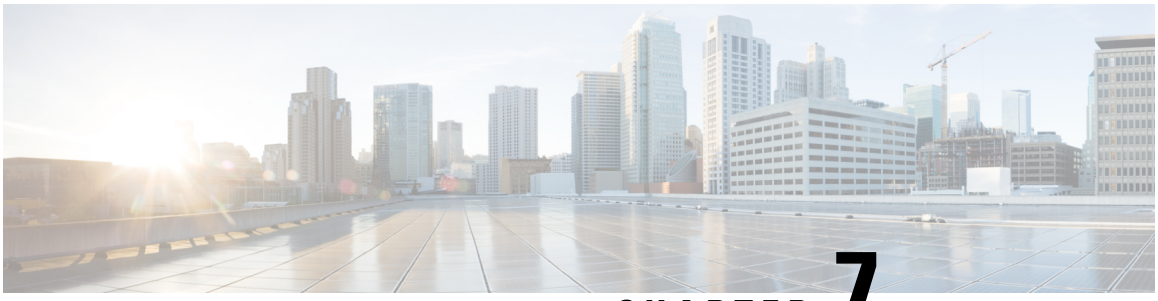
```
Router#show features acl
Fri June 3 19:17:31.635 UTC
Key:
```

```
X - Unsupported
- - Supported
? - Support unknown (optional package not installed)
* - Support data not available

[-] Access Control List (ACL)
|--[-]  IPv6 ACL Support
|   |--[*] IPv6 ACL ABF Track
|   |--[*] IPv6 ACL BNG
|   |--[*] IPv6 ACL Chaining (Meta ACL)
|   |--[-] IPv6 ACL Common ACL
|   |--[-] IPv6 ACL Compression
|   |--[*] IPv6 ACL Default ABF
|   |--[*] IPv6 ACL Fragment
|   |--[-] IPv6 ACL ICMP Off
|   |--[-] IPv6 ACL ICMP Protocol
|   |--[-] IPv6 ACL Interface Statistics
|   |--[-] IPv6 ACL Log Rate
|   |--[-] IPv6 ACL Log Threshold
|   |--[-] IPv6 ACL Logging
|   |--[-] IPv6 ACL MIB
|   |--[-] IPv6 ACL Object Groups (Scale)
|   |--[-] IPv6 ACL Police
|   |--[-] IPv6 ACL Priority
|   |--[*] IPv6 ACL Protocol Range
|   |--[-] IPv6 ACL Set Qos-Group
|   |--[-] IPv6 ACL Set TTL
|   |--[-] IPv6 ACL TCP Flags
|   |--[-] IPv6 ACL TTL Match
|   |--[-] IPv6 ACL UDF
|--[-] ES-ACL Support (L2 ACL)
|--[-] IPv4 ACL Support
|   |--[-] IPv4 ACL  Set Qos-group
|   |--[*] IPv4 ACL ABF Track
|   |--[*] IPv4 ACL BNG
|   |--[*] IPv4 ACL Chaining (Meta ACL)
|   |--[-] IPv4 ACL Common ACL
|   |--[-] IPv4 ACL Compression
|   |--[*] IPv4 ACL Default ABF
|   |--[*] IPv4 ACL Fragment
|   |--[-] IPv4 ACL Fragment Flags
|   |--[-] IPv4 ACL ICMP Off
|   |--[-] IPv4 ACL ICMP Protocol
|   |--[-] IPv4 ACL Interface Statistics
|   |--[-] IPv4 ACL Log Rate
|   |--[-] IPv4 ACL Log Threshold
|   |--[-] IPv4 ACL Logging
|   |--[-] IPv4 ACL MIB
|   |--[-] IPv4 ACL Object Groups (Scale)
|   |--[-] IPv4 ACL Police
|   |--[-] IPv4 ACL Priority
|   |--[*] IPv4 ACL Protocol Range
|   |--[-] IPv4 ACL Set TTL
|   |--[-] IPv4 ACL TCP Flags
|   |--[-] IPv4 ACL TTL
|   |--[-] IPv4 ACL UDF
|--[-] IPv4 Prefix-List
|--[-] IPv6 Prefix-List
```

View the List of Supported ACL Features for Specific RP

In this example, the capabilities for ACL features on the RP location `0/RP0/CPU0` are displayed.

```
Router#show features acl detail location 0/RP0/CPU0
Fri June 3 19:15:49.889 UTC
Key:
X - Unsupported
- - Supported
? - Support unknown (optional package not installed)
* - Support data not available

[-] Access Control List (ACL)
    Cisco provides basic traffic filtering capabilities with access control
    lists (also referred to as access lists). User can configure access
    control lists (ACLs) for all routed network protocols to filter protocol
    packets when these packets pass through a device. User can configure
    access lists on your device to control access to a network, access lists
    can prevent certain traffic from entering or exiting a network.
|--[-]  IPv6 ACL Support
|       IPv6 based ACL is a list of source IPv6 addresses that use Layer 3 or
|       Layer 4 information to permit or deny access to traffic. IPv6 router
|       ACLs apply only to IPv6 packets that are routed.. A filter contains the
|       rules to match the packet matches, the rule also stipulates if the
|       packet should be permitted or denied.
|   |--[*] IPv6 ACL ABF Track
|   |      IPv6 ACL ABF Track allows the user to configure a rule with track as
|   |      nexthop inside the ACL rule . ACL Based Forwarding (ABF) denotes the
|   |      ability to forward packets to another next hop router based on the
|   |      criteria defined in the rule. Track takes precedence over VRF and
|   |      IP, if present in the nexthop
|   |--[*] IPv6 ACL BNG
|   |      IPv6 ACL BNG is an ACL subscriber BNG feature. It allows the use of
|   |      ACL on dynamic template.
|   |--[*] IPv6 ACL Chaining (Meta ACL)
|   |      IPv6 ACL Chaining (Meta ACL) allows the user to apply more than one
|   |      ACL on the interface.  is known as Meta ACL or ACL chaining.
|   |--[-] IPv6 ACL Common ACL
|   |      IPv6 ACL Common allows the user to apply the ACL on the interface
|   |      using the common keyword. Using this feature the ACL won't be
|   |      applied to the specific interface but it will be common to th entire
|   |      NPU to which the interface belongs.
|   |--[-] IPv6 ACL Compression
|   |      IPv6 ACL Compression allows the user to apply the ACL on the
|   |      interface using a compression level. This helps in reducing the
|   |      hardware resources needed to program the ACL.
|   |--[*] IPv6 ACL Default ABF
|   |      IPv6 ACL Default ABF allows the user to configure a rule with
|   |      default nexthop inside the ACL rule . ACL Based Forwarding (ABF)
|   |      denotes the ability to forward packets to another next hop router
|   |      based on the criteria defined in the rule
|   |--[*] IPv6 ACL Fragment
|   |      IPv6 ACL Fragment allows the user to configure a rule with fragment
|   |      inside the ACL rule and use it as a match criteria to filter traffic.
|   |--[-] IPv6 ACL ICMP Off
|   |      IPv6 ACL ICMP Off allows the user to not genearte the ICMP error
|   |      message on a deny action. When configured it will not send the
|   |      packet to FIB to generate ICMP error message.
-------------------------- Truncated for Brevity --------------------------
```

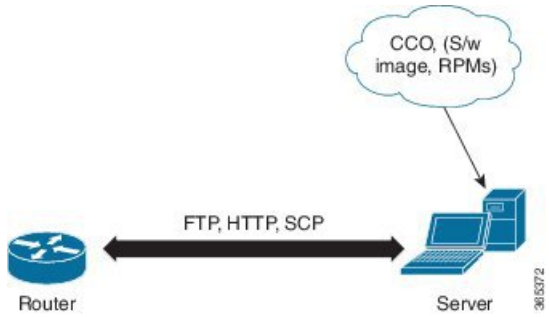


CHAPTER 7

Manage Automatic Dependency

Flexible packaging supports automatic dependency management. While you update an RPM, the system automatically identifies all relevant dependent packages and updates them.

Figure 5: Flow for Installation (base software, RPMs and SMUs)



Until this release, you downloaded the software image and required RPMs from CCO on a network server (the repository), and used the **install add** and the **install activate** commands to add and activate the downloaded files on the router. Then, you manually identify relevant dependent RPMs, to add and activate them.

With automatic dependency management, you need not identify dependent RPMs to individually add and activate them. You can execute new install command to identify and install dependent RPMs automatically.

The command **install source** adds and activates packages. The command **install replace** adds and activates packages in a given golden ISO (GISO).

Note

1. Cisco IOS XR Version 6.1.1 does not provide third party SMUs as part of automatic dependency management (**install source** command). The third party SMUs must be installed separately, and in isolation from other installation procedures (installation of SMUs and RPMs in IOS XR or admin containers).

The rest of this chapter contains these sections:

Update RPMs and SMUs

An RPM may contain a fix for a specific defect, and you may need to update the system with that fix. To update RPMs and SMUs to a newer version, use the **install source** command. When this command is issued for a particular RPM, the router communicates with the repository, and downloads and activates that RPM. If the repository contains a dependent RPM, the router identifies that dependent RPM and installs that too.

The syntax of the **install source** command is:

install source *repository* **[rpm]**

Four scenarios in which you can use the **install source** command are:

- **When a package name is not specified**

  When no package is specified, the command updates the latest SMUs of all installed packages.

  ```
  install source [repository]
  ```

- **When a package name is specified**

  If the package name is specified, the command installs that package, updates the latest SMUs of that package, along with its dependencies. If the package is already installed, only the SMUs of that package are installed. (SMUs that are already installed are skipped.)

  ```
  install source [repository] asr9k-mpls.rpm
  ```

- **When a package name and version number are specified**

  If a particular version of package needs to be installed, the complete package name must be specified; that package is installed along with the latest SMUs of that package present in the repository.

  ```
  install source [repository] asr9k-mpls-1.0.2.0-r710.x86_64.rpm
  ```

- **When an SMU is specified**

  If an SMU is specified, that SMU is downloaded and installed, along with its dependent SMUs.

  ```
  install source [repository] asr9k-mpls-1.0.2.1-r611.CSCub12345.x86_64.rpm
  ```

Upgrade Base Software Version

You can upgrade to a newer version of the base software when it becomes available. To upgrade to the latest base software version, use the **install source** command. With the upgrade of the base version, RPMs that are currently available on the router are also upgraded.

Note SMUs are not upgraded as part of this process.

The syntax of the **install source** command is:

install source *repository*

Note VRF and TPA on dataport is not supported. If the server is reachable only through non-default VRF interface, the file must already be retrieved using ftp, sfp, scp, http or https protocols.

Note Default routes (0.0.0.0/0) cannot be copied onto Linux due to TPA implementation.

You can use the **install source** command when:

- **The version number is specified**

  The base software (.mini) is upgraded to the specified version; all installed RPMs are upgraded to the same release version.

  install source *[repository]* **version** *<version>* **asr9k-mini-x64-<version>.iso**

  For example,

  install source *repository* **version 7.0.1 asr9k-mini-x64-7.0.1.iso**

  You can also automatically fetch the .mini file and RPMs of the required release and proceed with the upgrade.

  install source *repository* **asr9k-mini-x64-7.0.1.iso**

- **The version number for an RPM is specified**

  When performing a system upgrade, the user can choose to have an optional RPM to be of a different release (from that of the base software version); that RPM can be specified.

  install source *repository* **version 7.0.1 asr9k-mpls-1.0.2.0-r701.x86_64.rpm**

Downgrade an RPM

An RPM can be downgraded after it is activated. RPMs are of the following types:

- **Hostos RPM:** The RPM contains `hostos` in the name.

  For example:

  - <platform>-sysadmin-hostos-6.5.1-r651.CSChu77777.host.arm
  - <platform>-sysadmin-hostos-6.5.1-r651.CSChu77777.admin.arm
  - <platform>-sysadmin-hostos-6.5.1-r651.CSChu77777.host.x86_64
  - <platform>-sysadmin-hostos-6.5.1-r651.CSChu77777.admin.x86_64

- **Non-hostos RPM:** The RPM does not contain `hostos` in the name.

  For example:

  - <platform>-sysadmin-system-6.5.1-r651.CSCvc12346

To deactivate the RPMs, perform the following steps:

- **Downgrade Hostos RPM**
  - Scenario 1: To downgrade to version 06 from the active version 09:
    1. Download the version 06 hostos RPMs, and add the RPMs.

       ```
       install add source [repository]
       <platform>-sysadmin-hostos-6.5.1.06-r65108I.CSChu44444.host.arm
       <platform>-sysadmin-hostos-6.5.1.06-r65108I.CSChu44444.admin.arm
       <platform>-sysadmin-hostos-6.5.1.06-r65108I.CSChu44444.host.x86_64
       <platform>-sysadmin-hostos-6.5.1.06-r65108I.CSChu44444.admin.x86_64
       ```

    2. Activate the downloaded RPMs.

       ```
       install activate [repository]
       <platform>-sysadmin-hostos-6.5.1.06-r65108I.CSChu44444.host.arm
       <platform>-sysadmin-hostos-6.5.1.06-r65108I.CSChu44444.admin.arm
       <platform>-sysadmin-hostos-6.5.1.06-r65108I.CSChu44444.host.x86_64
       <platform>-sysadmin-hostos-6.5.1.06-r65108I.CSChu44444.admin.x86_64
       ```

    3. Commit the configuration.

       ```
       install commit
       ```

  - Scenario 2: Deactivate hostos RPM by activating base RPM, consider version 09 is active:
    1. Activate the base RPM.

       ```
       install activate <platform>-sysadmin-hostos-6.5.1.08I-r65108I.admin.arm
       <platform>-sysadmin-hostos-6.5.1.08I-r65108I.host.arm
       <platform>-sysadmin-hostos-6.5.1.08I-r65108I.admin.x86_64
       <platform>-sysadmin-hostos-6.5.1.08I-r65108I.host.x86_64
       ```

       For example, if RPM is the RPM installed, then is its base RPM.
    2. Commit the configuration.

       ```
       install commit
       ```

    The downgrade for third-party RPMs is similar to the hostos RPMs. To downgrade a SMU, activate the lower version of the SMU. If only one version of SMU is present, the base RPM of the SMU must be activated.

Note Hostos and third-party RPMs cannot be deactivated. Only activation of different versions is supported.

- **Downgrade Non-Hostos RPM**
  1. Deactivate the RPM to downgrade to earlier version of RPM.

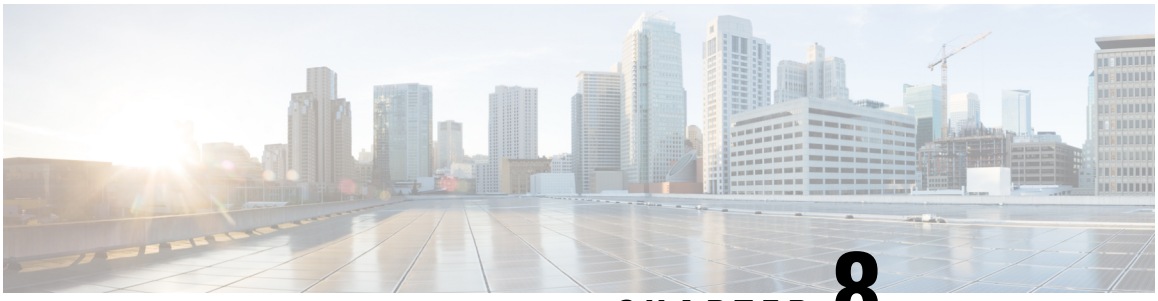
     ```
     install deactivate <platform>-<rpm-name>
     ```

  2. Check the active version of the RPM.

     ```
     show install active
     ```

  3. Commit the configuration.

```
install commit
```



CHAPTER 8

Customize Installation using Golden ISO

Table 6: Feature History Table

Feature Name	Release Information	Description
Automatic Install of Bridging Bug Fix RPMs	Release 7.5.1	This feature enables an easy one-step, no prompt upgrade, or downgrade, based on GISO. This removes the dependency on having to manually install the bridging bug fix RPMs before performing an upgrade or a downgrade.

Golden ISO (GISO) is a customized ISO that a user can build to suit the installation requirement. The user can customize the installable image to include the standard base image with the basic functional components, and add additional RPMs, SMUs and configuration files based on requirement.

The ease of installation and the time taken to seamlessly install or upgrade a system plays a vital role in a cloud-scale network. An installation process that is time-consuming and complex affects the resiliency and scale of the network. The GISO simplifies the installation process, automates the installation workflow, and manages the dependencies in RPMs and SMUs automatically.

GISO is built using a build script `gisobuild.py` available on the github location Github location.

From Cisco IOS XR Release 7.5.1, you can use the Automatic Install of Bridging Bug Fix RPMs feature to install the bridging bug fix RPMs that are prerequisite for a system upgrade or a downgrade. You need to add the required Bridging Bug Fix RPMs into the customized ISO built using Cisco Golden ISO (GISO) build script **gisobuild.py**. The GISO can include bridging Bug Fix RPMs for multiple releases, and installs only the specific bridging Bug Fix RPMs required for the target release. The bridging bug fix RPMs can be used in the following scenarios:

- To resolve a bug that might stop upgrade.
- The latest version has new prerequisite requirements that are not met by the earlier version.

When a system boots with GISO, additional SMUs and RPMs in GISO are installed automatically, and the router is pre-configured with the XR configuration in GISO. For more information about downloading and installing GISO, see Install Golden ISO, on page 85.

The capabilities of GISO can be used in the following scenarios:

- Migration from IOS XR 32-bit to IOS XR 64-bit

- Initial deployment of the router
- Software disaster recovery
- System upgrade from one base version to another
- System upgrade from same base version but with additional SMUs
- Install update to identify and update dependant packages

- Limitations, on page 74
- Customize Installation using Golden ISO, on page 73
- Golden ISO Workflow, on page 74
- Build Golden ISO, on page 75
- Install Golden ISO, on page 85

Limitations

The following are the known problems and limitations with the customized ISO:

- GISO image size more than 1.8 GB is not supported. The maximum image size for RSP880-LT-SE/TR is 1.599 GB.
- Building and booting GISO for asynchronous package (a package of different release than the ISO) is not supported.
- Verifying the XR configuration is not supported in the GISO build script `gisobuild.py`.
- Renaming a GISO build and then installing from the renamed GISO build is not supported.
- Install operation over IPv6 is not supported.
- Migrating from IOS XR 32-bit to 64-bit OS using GISO involves the following restrictions:
  - The IOS XR 32-bit to 64-bit conversion script does not support file names exceeding 48 characters.
  - The IOS XR 32-bit OS has a maximum file size limit of 2 GB. Ensure that GISO does not exceed that limit.

    For more information about migration methods and system requirements, see the Migration Guide for Cisco ASR 9000 Series Routers.

Customize Installation using Golden ISO

Table 7: Feature History Table

Feature Name	Release Information	Description
Automatic Install of Bridging Bug Fix RPMs	Release 7.5.1	This feature enables an easy one-step, no prompt upgrade, or downgrade, based on GISO. This removes the dependency on having to manually install the bridging bug fix RPMs before performing an upgrade or a downgrade.

Golden ISO (GISO) is a customized ISO that a user can build to suit the installation requirement. The user can customize the installable image to include the standard base image with the basic functional components, and add additional RPMs, SMUs and configuration files based on requirement.

The ease of installation and the time taken to seamlessly install or upgrade a system plays a vital role in a cloud-scale network. An installation process that is time-consuming and complex affects the resiliency and scale of the network. The GISO simplifies the installation process, automates the installation workflow, and manages the dependencies in RPMs and SMUs automatically.

GISO is built using a build script `gisobuild.py` available on the github location Github location.

From Cisco IOS XR Release 7.5.1, you can use the Automatic Install of Bridging Bug Fix RPMs feature to install the bridging bug fix RPMs that are prerequisite for a system upgrade or a downgrade. You need to add the required Bridging Bug Fix RPMs into the customized ISO built using Cisco Golden ISO (GISO) build script **gisobuild.py**. The GISO can include bridging Bug Fix RPMs for multiple releases, and installs only the specific bridging Bug Fix RPMs required for the target release. The bridging bug fix RPMs can be used in the following scenarios:

- To resolve a bug that might stop upgrade.
- The latest version has new prerequisite requirements that are not met by the earlier version.

When a system boots with GISO, additional SMUs and RPMs in GISO are installed automatically, and the router is pre-configured with the XR configuration in GISO. For more information about downloading and installing GISO, see Install Golden ISO, on page 85.

The capabilities of GISO can be used in the following scenarios:

- Migration from IOS XR 32-bit to IOS XR 64-bit
- Initial deployment of the router
- Software disaster recovery
- System upgrade from one base version to another
- System upgrade from same base version but with additional SMUs
- Install update to identify and update dependant packages

Limitations

The following are the known problems and limitations with the customized ISO:

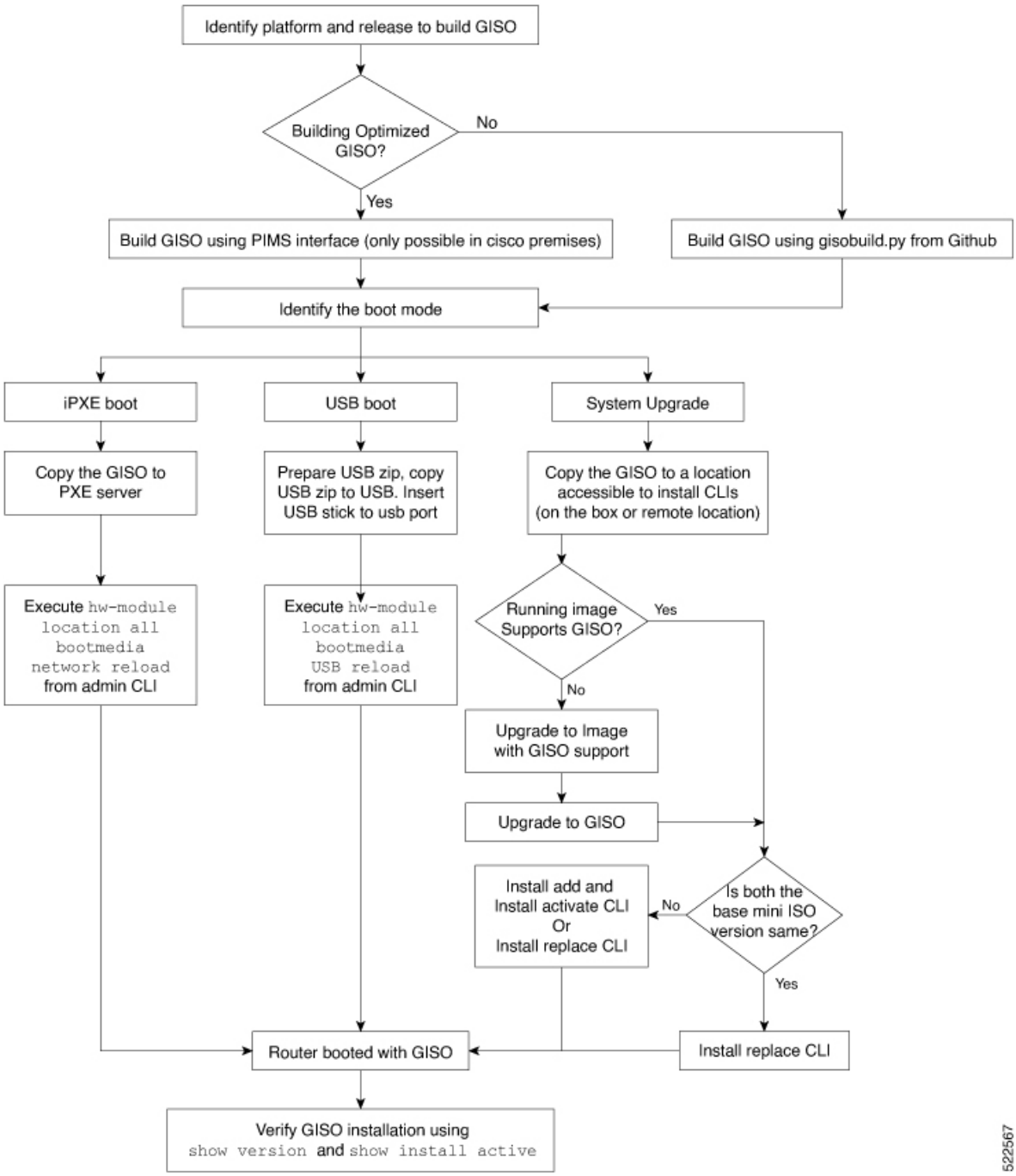
- GISO image size more than 1.8 GB is not supported. The maximum image size for RSP880-LT-SE/TR is 1.599 GB.
- Building and booting GISO for asynchronous package (a package of different release than the ISO) is not supported.
- Verifying the XR configuration is not supported in the GISO build script `gisobuild.py`.
- Renaming a GISO build and then installing from the renamed GISO build is not supported.
- Install operation over IPv6 is not supported.
- Migrating from IOS XR 32-bit to 64-bit OS using GISO involves the following restrictions:
  - The IOS XR 32-bit to 64-bit conversion script does not support file names exceeding 48 characters.
  - The IOS XR 32-bit OS has a maximum file size limit of 2 GB. Ensure that GISO does not exceed that limit.

    For more information about migration methods and system requirements, see the Migration Guide for Cisco ASR 9000 Series Routers.

Golden ISO Workflow

The following image shows the workflow for building and installing golden ISO.

Figure 6: Golden ISO Workflow



Build Golden ISO

The customized ISO is built using Cisco Golden ISO (GISO) build script `gisobuild.py` available on the Github location.

The GISO build script supports automatic dependency management, and provides these functionalities:

- Builds RPM database of all the packages present in package repository.
- Scans the repositories and selects the relevant Cisco RPMs that matches the input iso.
- Skips and removes third-party RPMs that are not SMUs of already existing third-party base package in mini-x.iso.
- Displays an error and exits build process if there are multiple base RPMs of same release but different versions.
- Performs compatibility check and dependency check for all the RPMs. For example, the child RPM asr9k-mpls-te-rsvp is dependent on the parent RPM asr9k-mpls. If only the child RPM is included, the Golden ISO build fails.

Build Golden ISO Using Script

Table 8: Feature History Table

Feature Name	Release Information	Description
Enhanced Golden ISO Build Tool	Release 7.5.1	This enhancement provides you with the flexibility to use the `gisobuild.py` tool to build GISO images using Cisco IOS XR software commands, YAML-based template file, or docker capability to suit your customized install requirements. When you build a GISO, you can also specify Zero Touch Provisioning (ZTP) initialization file, script initialization file, Cisco IOS XR configuration file, and SMUs in addition to using the base image and optional RPMs to automatically provision the router.

To build GISO, provide the following input parameters to the script:

- Base mini-x.iso (mandatory)
- XR configuration file (optional)
- one or more Cisco-specific SMUs for host, XR and System admin (optional)
- one or more third-party SMUs for host, XR and System admin (optional)
- Label for golden ISO (optional)
- Optional RPMs
- ZTP initialization `ztp.ini` file (optional)

- Script initialization `script.ini` file (optional)

The GISO script does not support verification of XR configuration.

Note To successfully add k9sec RPM to GISO, change the permission of the file to `644` using the **chmod** command.

```
chmod 644 [k9 sec rpm]
```

Cisco IOS XR, Release 7.5.1 introduces enhancements to the `gisobuild.py` GISO build tool. You can also add a `ztp.ini` ZTP initialization and `script.ini` Script initialization file. The ZTP configuration is applied on the router when the current software version is replaced or rolled back to a version with GISO image, and is used whenever ZTP is run to automatically provision the router. The tool supports more than one repository. You can use CLI command, docker, or a YAML file to build GISO.

Note

- Set the `migration` option to `true` when migrating from IOS XR 32-bit to IOS XR 64-bit software for Cisco ASR 9000 series routers.
- Set the `clean` option to `true` if you use the same build directory after the first GISO is created. Ensure that you set the option to `true` for every successive GISO build.
- Set the `docker` option to `true` if you are building GISO using docker.
- Ensure that the format and syntax of the YAML file is intact to avoid errors when building a GISO. For example, if the `:` symbol is missing, or if an unsupported symbol is used in the template, the GISO build displays errors.

The `gisobuild.py` tool can be run either natively or on systems where docker service is enabled and has the ability to pull published docker images. Prefer building the image using the docker as it does not require additional privileges:.

Note The `full-iso` option is used to build a full ISO image `xrv9k-full-x-7.5.1.iso` specific to Cisco IOS XRv 9000 routers. Starting Cisco IOS XR, Release 7.8.1, the full ISO image must not be used to build GISO.

To build GISO, perform the following steps:

Before you begin

- The system where GISO is built must meet the following requirements:
  - System must have Python version 3.6 and later.
  - System must have free disk space of minimum 12 GB.
  - Verify that the Linux utilities `mount`, `rm`, `cp`, `umount`, `zcat`, `chroot`, `mkisofs` are present in the system. These utilities will be used by the script. Ensure privileges are available to execute all of these Linux commands. However, if you are using docker, these utilities are not required.
  - Kernel version of the system must be later than 3.16 or later than the version of kernel of Cisco ISO.

- Verify that a `libyaml rpm` supported by the Linux kernel is available to successfully `import yaml` in the tool.
- User should have proper permission for security rpm(k9sec-rpm) in rpm repository, else security rpm would be ignored for Golden ISO creation.

- The system from where the `gisobuild.py` script is executed must have root credentials. This is not mandatory if you are building the image within a docker container.
- We recommend that you perform a `git pull` operation before you use the `gisobuild.py` script to ensure you obtain the latest version of the script for the Python version.

Step 1 Copy the script `gisobuild.py` from the Github repository to an offline system or external server where the GISO will be built. Ensure that this system meets the pre-requisites described above in the *Before You Begin* section.

Step 2 Run the script `gisobuild.py` and provide parameters to build the golden ISO off the router. Ensure that all RPMs and SMUs are present in the same directory or on a repository. The number of RPMs and SMUs that can be used to build the Golden ISO is 64.

```
usage: gisobuild.py [-h] [--iso ISO] [--repo REPO [REPO ...]]
                    [--bridging-fixes BRIDGE_FIXES [BRIDGE_FIXES ...]]
                    [--xrconfig XRCONFIG] [--ztp-ini ZTP_INI] [--label LABEL]
                    [--out-directory OUT_DIRECTORY] [--yamlfile CLI_YAML] [--clean]
                    [--pkglist PKGLIST [PKGLIST ...]] [--script SCRIPT] [--docker]
                    [--x86-only] [--migration]
                    [--remove-packages REMOVE_PACKAGES [REMOVE_PACKAGES ...]]
                    [--skip-usb-image] [--copy-dir COPY_DIRECTORY]
                    [--clear-bridging-fixes] [--verbose-dep-check] [--debug]
                    [--version]

Utility to build Golden ISO for IOS-XR.

optional arguments:
  -h, --help            show this help message and exit
  --iso ISO             Path to Mini.iso/Full.iso file
  --repo REPO [REPO ...]
                        Path to RPM repository. For LNT, user can specify .rpm, .tgz,
                        .tar filenames, or directories. RPMs are only used if already
                        included in the ISO, or specified by the user via the
                        --pkglist option.
  --bridging-fixes BRIDGE_FIXES [BRIDGE_FIXES ...]
                        Bridging rpms to package. For EXR, takes from-release or rpm
                        names; for LNT, the user can specify the same file types as for
                        the --repo option.
  --xrconfig XRCONFIG   Path to XR config file
  --ztp-ini ZTP_INI     Path to user ztp ini file
  --label LABEL, -l LABEL
                        Golden ISO Label
  --out-directory OUT_DIRECTORY
                        Output Directory
  --yamlfile CLI_YAML   Cli arguments via yaml
  --clean               Delete output dir before proceeding
  --pkglist PKGLIST [PKGLIST ...]
                        Packages to be added to the output GISO. For eXR: optional rpm
                        or smu to package. For LNT: either full package filenames or
                        package names for user installable packages can be specified.
                        Full package filenames can be specified to choose a particular
                        version of a package, the rest of the block that the package is
                        in will be included as well. Package names can be specified to
                        include optional packages in the output GISO.
```

```
  --docker, --use-container
                        Build GISO in container environment.Pulls and run pre-built
                        container image to build GISO.
  --version             Print version of this script and exit

EXR only build options:
  --script SCRIPT       Path to user executable script executed as part of bootup post
                        activate.
  --x86-only            Use only x86_64 rpms even if other architectures are
                        applicable.
  --migration           To build Migration tar only for ASR9k

LNT only build options:
  --remove-packages REMOVE_PACKAGES [REMOVE_PACKAGES ...]
                        Remove RPMs, specified in a comma separated list. These are are
                        matched against user installable package names, and must be the
                        whole package name, e.g: xr-bgp
  --skip-usb-image      Do not build the USB image
  --copy-dir COPY_DIRECTORY
                        Copy built artefacts to specified directory if provided. The
                        specified directory must already exist, be writable by the
                        builder and must not contain a previously built artefact with
                        the same name.
  --clear-bridging-fixes
                        Remove all bridging bugfixes from the input ISO
  --verbose-dep-check   Verbose output for the dependency check.
  --debug               Output debug logs to console
```

Example

Example: Build Docker-Based GISO Image

In this example, a GISO image is built using docker.

```
[root@xr src]# ./gisobuild.py --docker --iso /auto/ncs5500giso/ncs5500-mini-x-7.5.1.iso
--repo /auto/ncs5500giso --pkglist ncs5500-bgp-2.0.0.0-r751.x86_64.rpm
ncs5500-eigrp-1.0.0.0-r751.x86_64.rpm ncs5500-isis-2.1.0.0-r751.x86_64.rpm
ncs5500-k9sec-3.1.0.0-r751.x86_64.rpm
ncs5500-li-1.0.0.0-r751.x86_64.rpm ncs5500-mcast-3.0.0.0-r751.x86_64.rpm
ncs5500-mgbl-3.0.0.0-r751.x86_64.rpm
ncs5500-mpls-2.1.0.0-r751.x86_64.rpm ncs5500-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
ncs5500-ospf-2.0.0.0-r751.x86_64.rpm
ncs5500-parser-2.0.0.0-r751.x86_64.rpm --label dockerbasedgiso

Local System requirements check [PASS]
Pulling gisobuild image from hub. Please wait...
\
Done...
System requirements check [PASS]

Platform: ncs5500 Version: 7.5.1

Scanning repository [/auto/ncs5500giso]...

Building RPM Database...

Total 11 RPM(s) present in the repository path provided in CLI
[ 1] ncs5500-mpls-2.1.0.0-r751.x86_64.rpm
[ 2] ncs5500-mgbl-3.0.0.0-r751.x86_64.rpm
[ 3] ncs5500-bgp-2.0.0.0-r751.x86_64.rpm
[ 4] ncs5500-parser-2.0.0.0-r751.x86_64.rpm
```

```
[ 5] ncs5500-isis-2.1.0.0-r751.x86_64.rpm
[ 6] ncs5500-mcast-3.0.0.0-r751.x86_64.rpm
[ 7] ncs5500-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
[ 8] ncs5500-ospf-2.0.0.0-r751.x86_64.rpm
[ 9] ncs5500-li-1.0.0.0-r751.x86_64.rpm
[10] ncs5500-eigrp-1.0.0.0-r751.x86_64.rpm
[11] ncs5500-k9sec-3.1.0.0-r751.x86_64.rpm

Following XR x86_64 rpm(s) will be used for building Golden ISO:

        (+) ncs5500-ospf-2.0.0.0-r751.x86_64.rpm
        (+) ncs5500-bgp-2.0.0.0-r751.x86_64.rpm
        (+) ncs5500-parser-2.0.0.0-r751.x86_64.rpm
        (+) ncs5500-mcast-3.0.0.0-r751.x86_64.rpm
        (+) ncs5500-li-1.0.0.0-r751.x86_64.rpm
        (+) ncs5500-eigrp-1.0.0.0-r751.x86_64.rpm
        (+) ncs5500-mgbl-3.0.0.0-r751.x86_64.rpm
        (+) ncs5500-mpls-2.1.0.0-r751.x86_64.rpm
        (+) ncs5500-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
        (+) ncs5500-isis-2.1.0.0-r751.x86_64.rpm
        (+) ncs5500-k9sec-3.1.0.0-r751.x86_64.rpm

        ...RPM signature check [PASS]

Skipping following rpms from repository since they are already present in base ISO:

        (-) ncs5500-parser-2.0.0.0-r751.x86_64.rpm
        (-) ncs5500-bgp-2.0.0.0-r751.x86_64.rpm

        ...RPM compatibility check [PASS]

Building Golden ISO...
Summary .....

XR rpms:
        ncs5500-mcast-3.0.0.0-r751.x86_64.rpm
        ncs5500-mgbl-3.0.0.0-r751.x86_64.rpm
        ncs5500-isis-2.1.0.0-r751.x86_64.rpm
        ncs5500-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
        ncs5500-eigrp-1.0.0.0-r751.x86_64.rpm
        ncs5500-mpls-2.1.0.0-r751.x86_64.rpm
        ncs5500-ospf-2.0.0.0-r751.x86_64.rpm
        ncs5500-li-1.0.0.0-r751.x86_64.rpm
        ncs5500-k9sec-3.1.0.0-r751.x86_64.rpm

        ...Golden ISO creation SUCCESS.

Golden ISO Image Location: /var/tmp/giso/gisobuild-toolkit-master/src/output_gisobuild/
                           ncs5500-golden-x-7.5.1-dockerbasedgiso.iso

[root@xr src]# ./gisobuild.py --docker --iso /auto/asr9kgiso/asr9k-mini-x-7.5.1.iso
--repo /auto/asr9kgiso --pkglist asr9k-7.5.1.CSCsb88888 asr9k-bgp-2.0.0.0-r751.x86_64.rpm
asr9k-eigrp-1.0.0.0-r751.x86_64.rpm asr9k-isis-2.1.0.0-r751.x86_64.rpm
asr9k-k9sec-3.1.0.0-r751.x86_64.rpm
asr9k-li-1.0.0.0-r751.x86_64.rpm asr9k-mcast-3.0.0.0-r751.x86_64.rpm
asr9k-mgbl-3.0.0.0-r751.x86_64.rpm
asr9k-mpls-2.1.0.0-r751.x86_64.rpm asr9k-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
asr9k-ospf-2.0.0.0-r751.x86_64.rpm
asr9k-parser-2.0.0.0-r751.x86_64.rpm --label dockerbasedgiso

Local System requirements check [PASS]
Pulling gisobuild image from hub. Please wait...
\
Done...
```

```
System requirements check [PASS]

Platform: asr9000 Version: 7.5.1

Scanning repository [/auto/asr9000giso]...

Building RPM Database...

Total 11 RPM(s) present in the repository path provided in CLI
[ 1] asr9k-mpls-2.1.0.0-r751.x86_64.rpm
[ 2] asr9k-mgbl-3.0.0.0-r751.x86_64.rpm
[ 3] asr9k-bgp-2.0.0.0-r751.x86_64.rpm
[ 4] asr9k-parser-2.0.0.0-r751.x86_64.rpm
[ 5] asr9k-isis-2.1.0.0-r751.x86_64.rpm
[ 6] asr9k-mcast-3.0.0.0-r751.x86_64.rpm
[ 7] asr9k-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
[ 8] asr9k-ospf-2.0.0.0-r751.x86_64.rpm
[ 9] asr9k-li-1.0.0.0-r751.x86_64.rpm
[10] asr9k-eigrp-1.0.0.0-r751.x86_64.rpm
[11] asr9k-k9sec-3.1.0.0-r751.x86_64.rpm
Following XR x86_64 rpm(s) will be used for building Golden ISO:

        (+) asr9k-ospf-2.0.0.0-r751.x86_64.rpm
        (+) asr9k-bgp-2.0.0.0-r751.x86_64.rpm
        (+) asr9k-parser-2.0.0.0-r751.x86_64.rpm
        (+) asr9k-mcast-3.0.0.0-r751.x86_64.rpm
        (+) asr9k-li-1.0.0.0-r751.x86_64.rpm
        (+) asr9k-eigrp-1.0.0.0-r751.x86_64.rpm
        (+) asr9k-mgbl-3.0.0.0-r751.x86_64.rpm
        (+) asr9k-mpls-2.1.0.0-r75114I.x86_64.rpm
        (+) asr9k-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
        (+) asr9k-isis-2.1.0.0-r751.x86_64.rpm
        (+) asr9k-k9sec-3.1.0.0-r751.x86_64.rpm

        ...RPM signature check [PASS]

Skipping following rpms from repository since they are already present in base ISO:

        (-) asr9k-parser-2.0.0.0-r751.x86_64.rpm
        (-) asr9k-bgp-2.0.0.0-r751.x86_64.rpm

        ...RPM compatibility check [PASS]

Building Golden ISO...
Summary .....

XR rpms:
        asr9k-mcast-3.0.0.0-r751.x86_64.rpm
        asr9k-mgbl-3.0.0.0-r751.x86_64.rpm
        asr9k-isis-2.1.0.0-r751.x86_64.rpm
        asr9k-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
        asr9k-eigrp-1.0.0.0-r751.x86_64.rpm
        asr9k-mpls-2.1.0.0-r751.x86_64.rpm
        asr9k-ospf-2.0.0.0-r751.x86_64.rpm
        asr9k-li-1.0.0.0-r751.x86_64.rpm
        asr9k-k9sec-3.1.0.0-r751.x86_64.rpm

        ...Golden ISO creation SUCCESS.

Golden ISO Image Location: /var/tmp/giso/gisobuild-toolkit-master/src/output_gisobuild/
                           asr9k-golden-x-7.5.1-dockerbasedgiso.iso
```

View that the GISO file is created succesfully.

```
[root@xr src]# ls
exrmod  gisobuild.py  lntmod  output_gisobuild  utils

[root@xr src]# cd output_gisobuild/
[root@xr output_gisobuild]# ls
img_built_name.txt logs ncs5500asr9k-golden-x-7.5.1-dockerbasedgiso.iso
rpms_packaged_in_giso.txt
```

Example: Build YAML-Based GISO Image

YAML is a markup file that serves as a template to provide the package list and manage the build options.

The following example shows a sample YAML template:

```
# Options below correspond to the tool input options.
# --iso ISO             Path to Mini.iso/golden.iso file
# --repo REPO [REPO ...]
#                       Path to list of RPM repositories. RPMs are only used if already
#                       included in the ISO, or specified by the user via the --pkglist
option.
# --pkglist PKGLIST [PKGLIST ...]
#                       Optional list of rpm or smu to add to the ISO.
# --remove-packages REMOVE_PACKAGES [REMOVE_PACKAGES ...]
#                      Remove named RPMs, specified in a space separated list. Valid build
#                       option for LNT only. eXR builds simply ignores this option.
# --bridging-fixes BRIDGE_FIXES [BRIDGE_FIXES ...]
#                       Bridging rpms to package. Takes from-release (supported for eXR)
#                       or rpm names.
# --xrconfig XRCONFIG   Path to XR config file
# --ztp-ini ZTP_INI     Path to user ztp ini file
# --script SCRIPT       Path to user executable script executed as part of
#                       bootup post activate. Valid build option for eXR only.
#                       LNT builds simply ignores.
# --label LABEL         Golden ISO Label
# --out-directory OUT_DIRECTORY
#                       Output Directory. Built GISO and logs will be available post
gisobuild.
# --copy-directory COPY_DIRECTORY
#                      Copy built artefacts to specified directory if provided. Valid build
#                       option for LNT only. eXR build ignores this option.
# --yamlfile CLI_YAML   Cli arguments via yaml.
# --clean               Delete output dir before proceeding.
# --migration           To build Migration tar only for ASR9k. Valid build option for eXR
only.
#                       LNT builds simply ignore this option.
# --docker              Load and run pre-built docker image. Valid build option for eXR
only.
#                       LNT builds simply ignore this option.
# --x86-only            Use only x86_64 rpms even if other architectures are applicable.
Valid build
#                       option for eXR only. LNT builds simply ignore this option.
# --version             Print version of this script and exit

packages:
    iso: <path-to-iso>
    repo:
        - <path-to-repo1>
        - <path-to-repo2>
    pkglist:
        - <pkg1>
        - <pkg2>
    bridge-fixes:
        upgrade-from-release:
            - <dotted-release-1>
```

```
            - <dotted-release-2>
        rpms:
            - <pkg1>
            - <pkg2>
    remove_packages:
        - <pkg1>
        - <pkg2>

user-content:
    script: <path-to-script-sh>
    xrconfig: <path-to-router.cfg>
    ztp-ini: <path-to-ztp.ini>

output:
    label: <giso-label>
    out-directory: <path-to-output-directory>
    clean: <true/false>

options:
    docker: <true/false>
    migration: <true/false>
    x86-only: <true/false>
```

In this example, you configure a YAML file with the required files:

```
packages:
    iso: /auto/751_repo/ncs5500-mini-x-7.5.1.iso
    repo:
        - /auto/751_repo/
    pkglist:
        - ncs5500-bgp-2.0.0.0-r751.x86_64.rpm
        - ncs5500-eigrp-1.0.0.0-r751.x86_64.rpm
        - ncs5500-isis-2.1.0.0-r751.x86_64.rpm
        - ncs5500-li-1.0.0.0-r751.x86_64.rpm
        - ncs5500-mcast-3.0.0.0-r751.x86_64.rpm
        - ncs5500-mgbl-3.0.0.0-r751.x86_64.rpm
        - ncs5500-mpls-2.1.0.0-r751.x86_64.rpm
        - ncs5500-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
        - ncs5500-ospf-2.0.0.0-r751.x86_64.rpm
        - ncs5500-parser-2.0.0.0-r751.x86_64.rpm
        - ncs5500-k9sec-3.1.0.0-r751.x86_64.rpm
        - ncs5500-mcast-3.0.0.1-r751.CSCxr33333.x86_64.rpm
        - ncs5500-os-5.0.0.1-r751.CSCxr11111.x86_64.rpm
        - ncs5500-sysadmin-hostos-7.5.1-r751.CSCho99999.admin.x86_64.rpm
        - ncs5500-sysadmin-hostos-7.5.1-r751.CSCho99999.host.x86_64.rpm
        - ncs5500-sysadmin-topo-7.5.1-r751.CSCcv55555.x86_64.rpm
        - ncs5500-sysadmin-system-7.5.1-r751.CSCcv44444.x86_64.rpm
        - openssh-scp-6.6p1.p1-r0.5.0.r751.CSCtp11111.xr.x86_64.rpm
        - cisco-klm-zermatt-0.1.p1-r0.0.r751.CSCtp11111.xr.x86_64.rpm

    remove_rpms: []

user-content:
    script: script.sh
    xrconfig: /auto/751_repo/gisoxrconfig.cfg
    ztp-ini:  /auto/751_repo/ztp.ini

output:
    label: 751_yaml_install
    out-directory: /auto/751_repo/
    clean: true

options:
    docker: false
    full-iso: false
```

```
    migration: false
    x86-only: false

packages:
    iso: /auto/751_repo/asr9k-mini-x-7.5.1.iso
    repo:
        - /auto/751_repo/
    pkglist:
        - asr9k-bgp-2.0.0.0-r751.x86_64.rpm
        - asr9k-eigrp-1.0.0.0-r751.x86_64.rpm
        - asr9k-isis-2.1.0.0-r751.x86_64.rpm
        - asr9k-li-1.0.0.0-r751.x86_64.rpm
        - asr9k-mcast-3.0.0.0-r751.x86_64.rpm
        - asr9k-mgbl-3.0.0.0-r751.x86_64.rpm
        - asr9k-mpls-2.1.0.0-r751.x86_64.rpm
        - asr9k-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
        - asr9k-ospf-2.0.0.0-r751.x86_64.rpm
        - asr9k-parser-2.0.0.0-r751.x86_64.rpm
        - asr9k-k9sec-3.1.0.0-r751.x86_64.rpm
        - asr9k-mcast-3.0.0.1-r751.CSCxr33333.x86_64.rpm
        - asr9k-os-5.0.0.1-r751.CSCxr11111.x86_64.rpm
        - asr9k-sysadmin-hostos-7.5.1-r751.CSCho99999.admin.x86_64.rpm
        - asr9k-sysadmin-hostos-7.5.1-r751.CSCho99999.host.x86_64.rpm
        - asr9k-sysadmin-topo-7.5.1-r751.CSCcv55555.x86_64.rpm
        - asr9k-sysadmin-system-7.5.1-r751.CSCcv44444.x86_64.rpm
        - openssh-scp-6.6p1.p1-r0.5.0.r751.CSCtp11111.xr.x86_64.rpm
        - cisco-klm-zermatt-0.1.p1-r0.0.r751.CSCtp11111.xr.x86_64.rpm

    remove_rpms: []

user-content:
    script: script.sh
    xrconfig: /auto/751_repo/gisoxrconfig.cfg
    ztp-ini:  /auto/751_repo/ztp.ini

output:
    label: 751_yaml_install
    out-directory: /auto/751_repo/
    clean: true

options:
    docker: false
    full-iso: false
    migration: false
    x86-only: false
```

If you do not want to specify the list of packages and parameters via CLI, you can use the YAML file template.

```
[directory-path]$ ./src/gisobuild.py --yamlfile <input-yaml-cfg>
```

To override any input in the YAML configuration file, use the corresponding CLI options.

```
[directory-path]$ ./src/gisobuild.py --yamlfile <input-yaml-cfg> --label <new-label>
```

This new label overrides the label specified in the YAML file.

When the host machine does not have its package dependencies met, but allows pulling and running docker images, enable the docker option in YAML file to `true` and run the command:

```
[directory-path]$ ./src/gisobuild.py --yamlfile <input-yaml-cfg>
```

where, the `input-yaml-cfg` has the docker option set to `true`.

What to do next

Install the GISO image on the router.

Install Golden ISO

Golden ISO (GISO) automatically performs the following actions:

- Installs host and system admin RPMs.
- Partitions repository and TFTP boot on RP.
- Creates software profile in system admin and XR modes.
- Installs XR RPMs. Use **show install active** command to see the list of RPMs.
- Applies XR configuration. Use **show running-config** command in XR mode to verify.

Step 1 Download GISO image to the router using one of the following options:

- **PXE boot:** when the router is booted, the boot mode is identified. After detecting PXE as boot mode, all available ethernet interfaces are brought up, and DHClient is run on each interface. DHClient script parses HTTP or TFTP protocol, and GISO is downloaded to the box.

  When you bring up a router using the PXE boot mode, existing configurations are removed. To recover smart licensing configurations like Permanent License Reservation (PLR), enable these configurations after the router comes up.

```
Router#configure
Router(config)#license smart reservation
Router(config)#commit
```

  The following example shows the logs from installation of GISO using PXE boot:

```
...
Fri Dec  2 19:18:03 UTC 2016: ---Starting to prepare host logical volume---
...
Fri Dec 02 19:18:14 UTC 2016: Skipping tp base rpm(openssh-scp-6.6p1-r0.0.host.x86_64.rpm) from
 installation
Fri Dec 02 19:18:14 UTC 2016: Skipping tp base rpm(kernel-modules-3.14-r0.1.host.x86_64.rpm) from
 installation
Fri Dec 02 19:18:15 UTC 2016: Installing asr9k-sysadmin-hostos-6.1.3-r613.CSChu77777.host.x86_64
           [SUCCESS]

...

Fri Dec  2 19:18:23 UTC 2016: ---Starting to prepare calvados logical volume---

...

Fri Dec 02 19:18:48 UTC 2016: Skipping tp base rpm(openssh-scp-6.6p1-r0.0.admin.x86_64.rpm) from
 installation
Fri Dec 02 19:18:48 UTC 2016: Skipping tp base rpm(kernel-modules-3.14-r0.1.admin.x86_64.rpm)
from installation
Fri Dec 02 19:18:49 UTC 2016: Installing asr9k-sysadmin-system-6.1.3-r613.CSCcv44444.x86_64
            [SUCCESS]
Fri Dec 02 19:18:50 UTC 2016: Installing asr9k-sysadmin-shared-6.1.3-r613.CSCcv33333.x86_64
            [SUCCESS]
Fri Dec 02 19:18:51 UTC 2016: Installing asr9k-sysadmin-hostos-6.1.3-r613.CSChu77777.admin.x86_64
```

```
          [SUCCESS]

...

Fri Dec  2 19:19:07 UTC 2016: ---Starting to prepare repository---
Fri Dec  2 19:19:11 UTC 2016: File system creation on /test took 3 seconds
Fri Dec  2 19:19:11 UTC 2016: Copying /iso/host.iso to repository /iso directory
Fri Dec  2 19:19:11 UTC 2016: Copy Host rpms to repository
Fri Dec  2 19:19:13 UTC 2016: Copying /iso/asr9k-sysadmin.iso to repository /iso directory
Fri Dec  2 19:19:13 UTC 2016: Copy Sysadmin rpms to repository
Fri Dec  2 19:19:16 UTC 2016: Copy HostOs rpms to repository
Fri Dec  2 19:19:16 UTC 2016: Copy XR rpms to repository
Fri Dec  2 19:19:16 UTC 2016: Copy giso_info.txt to repository
Fri Dec  2 19:19:17 UTC 2016: Copying /iso/asr9k-xr.iso to repository /iso directory
Fri Dec  2 19:19:21 UTC 2016: Copying all ISOs to repository took 10 seconds

...
```

- **USB boot or Disk Boot:** when the USB mode is detected during boot, and GISO is identified, the additional RPMs and XR configuration files are extracted and installed.
- **System Upgrade:** when the system is upgraded, GISO can be installed using **install add**, **install activate**, or using **install replace** commands.

Important To replace the current version and packages on the router with the version from GISO, note the change in command and format.

- In versions prior to Cisco IOS XR Release 6.3.3, 6.4.x and 6.5.1, use the **install update** command:

  ```
  install update source <source path> <Golden-ISO-name> replace
  ```

- In Cisco IOS XR Release 6.5.2 and later, use the **install replace** command.

  ```
  install replace <absolute-path-of-Golden-ISO>
  ```

Note To create a Bootable External USB Disk, do the following:

- Ensure that the USB Boot Disk has a minimum storage of 8GB, and that you have root/admin or appropriate permission to create bootable disk on linux machine.

a. Copy and execute usb-install script on the Linux machine to create a bootable external USB.

```
Router#admin

sysadmin-vm:0_RSP0#run chvrf 0 ssh rp0_admin
[sysadmin-vm:0_RSP0:~]$ssh my_host
[host:~]$cd /misc/disk1/
[host:~]$./usb-install-712-or-latest.sh asr9k-goldenk9-x64-7.0.2-dr.iso /dev/sdc
EFI

Preparing USB stick for EFI
parted gpt: Failed to create partition - continuing ...
Create filesystem on /dev/sdc1
Mounting source iso at //misc/disk1/cdtmp.CnuKnA
Mounting destination /dev/sdc1 at //misc/disk1/usbdev.SSBb4R
Copying image to USB stick
Initrd path is //misc/disk1/cdtmp.CnuKnA/boot/initrd.img
Getting boot
3749342 blocks
Copying boot
Copying initrd.img
Copying signature.initrd.img
Copying certs
Creating grub files
Copying /misc/disk1/asr9k-goldenk9-x64-7.0.2-dr.iso in USB Stick
USB stick set up for EFI boot!
```

b. Reset the RSP/RP and plug in bootable USB to RSP/RP's front panel. The USB will get detected in ROMMON. Note that when the system is in ROMMON, and if you add a front panel external USB, the USB will not be detected until the RSP/RP is reset.

The options to upgrade the system are as follows:

- **system upgrade from a non-GISO (image that does not support GISO) to GISO image:** If a system is running a version1 with an image that does not support GISO, the system cannot be upgraded directly to version2 of an image that supports GISO. Instead, the version1 must be upgraded to version2 mini ISO, and then to version2 GISO.
- **system upgrade in a release from version1 GISO to version2 GISO:** If both the GISO images have the same base version but different labels, **install add** and **install activate** commands does not support same version of two images. Instead, using **install source** command installs only the delta RPMs. System reload is based on restart type of the delta RPMs.

  Using **install replace** command performs a system reload, irrespective of the difference between ISO and the existing version.
- **system upgrade across releases from version1 GISO to version2 GISO:** Both the GISO images have different base versions. Use **install add** and **install activate** commands, or **install replace** command to perform the system upgrade. The router reloads after the upgrade with the version2 GISO image.

Step 2 Run the **show install repository all** command in System Admin mode to view the RPMs and base ISO for host, system admin and XR.

```
sysadmin-vm:0_RP0#show install repository all
Admin repository
---------------------
 asr9k-sysadmin-6.1.1

 asr9k-sysadmin-hostos-6.1.1-r611.CSCcv10001.admin.x86_64
 asr9k-sysadmin-system-6.1.1-r611.CSCcv10005.x86_64
....

  XR repository
------------------
 asr9k-iosxr-mgbl-3.0.0.0-r611.x86_64
 asr9k-xr-6.1.1
....

  Host repository
---------------------
 host-6.1.1
```

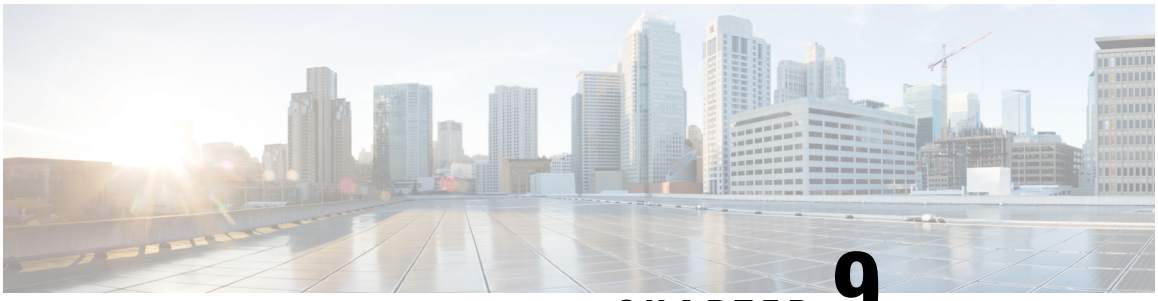
Step 3 Run the **show install package <golden-iso>** command to display the list of RPMs, and packages built in GISO.

Note To list RPMs in the GISO, the GISO must be present in the install repository.

```
Router#show install package asr9k-goldenk9-x64-6.1.3

Sun Dec  4 13:52:48.279 UTC
This may take a while ...
    ISO Name: asr9k-goldenk9-x64-6.1.3
    ISO Type: bundle
    ISO Bundled: asr9k-mini-x64-6.1.3
    Golden ISO Label: temp
    ISO Contents:
        ISO Name: asr9k-xr-6.1.3
        ISO Type: xr
        rpms in xr ISO:
            iosxr-os-asr9k-64-5.0.0.0-r613
            iosxr-ce-asr9k-64-3.0.0.0-r613
            iosxr-infra-asr9k-64-4.0.0.0-r613
            iosxr-fwding-asr9k-64-4.0.0.0-r613
            iosxr-routing-asr9k-64-3.1.0.0-r613
        ...

        ISO Name: asr9k-sysadmin-6.1.3
        ISO Type: sysadmin
        rpms in sysadmin ISO:
            asr9k-sysadmin-topo-6.1.3-r613
            asr9k-sysadmin-shared-6.1.3-r613
            asr9k-sysadmin-system-6.1.3-r613
            asr9k-sysadmin-hostos-6.1.3-r613.admin
        ...

        ISO Name: host-6.1.3
        ISO Type: host
        rpms in host ISO:
            asr9k-sysadmin-hostos-6.1.3-r613.host

        Golden ISO Rpms:
            xr rpms in golden ISO:
                asr9k-k9sec-x64-2.2.0.1-r613.CSCxr33333.x86_64.rpm
                openssh-scp-6.6p1.p1-r0.0.CSCtp12345.xr.x86_64.rpm
                openssh-scp-6.6p1-r0.0.xr.x86_64.rpm
                asr9k-mpls-x64-2.1.0.0-r613.x86_64.rpm
                asr9k-k9sec-x64-2.2.0.0-r613.x86_64.rpm

            sysadmin rpms in golden ISO:
```

```
    asr9k-sysadmin-system-6.1.3-r613.CSCcv11111.x86_64.rpm
    openssh-scp-6.6p1-r0.0.admin.x86_64.rpm
    openssh-scp-6.6p1.p1-r0.0.CSCtp12345.admin.x86_64.rpm

host rpms in golden ISO:
    openssh-scp-6.6p1-r0.0.host.x86_64.rpm
    openssh-scp-6.6p1.p1-r0.0.CSCtp12345.host.x86_64.rpm
```

The ISO, SMUs and packages in GISO are installed on the router.



CHAPTER 9

Deploy Router Using Classic ZTP

Manually deploying network devices in a large-scale environment requires skilled workers and is time consuming.

With Zero Touch Provisioning (ZTP), you can seamlessly provision thousands of network devices accurately within minutes and without any manual intervention. This can be easily defined using a configuration file or script using shell or phyton.

ZTP provides multiple options, such as:

- Automatically apply specific configuration in a large-scale environment.
- Download and install specific IOS XR image.
- Install specific application package or third party applications automatically.
- Deploy containers without manual intervention.
- Upgrade or downgrade software versions effortlessly on thousands of network devices at a time

Benefits of Using ZTP

ZTP helps you manage large-scale service providers infrastructures effortlessly. Following are the added benefits of using ZTP:

- ZTP helps you to remotely provision a router anywhere in the network. Thus eliminates the need to send an expert to deploy network devices and reduces IT cost.
- Automated provisioning using ZTP can remove delay and increase accuracy and thus is cost-effective and provides better customer experience.

  By automating repeated tasks, ZTP allows network administrators to concentrate on more important stuff.
- ZTP process helps you to quickly restore service. Rather than troubleshooting an issue by hand, you can reset a system to well-known working status.

Use Cases

The following are some of the useful use cases for ZTP:

- Using ZTP to install Chef
- Using ZTP to integrate IOS-XR with NSO

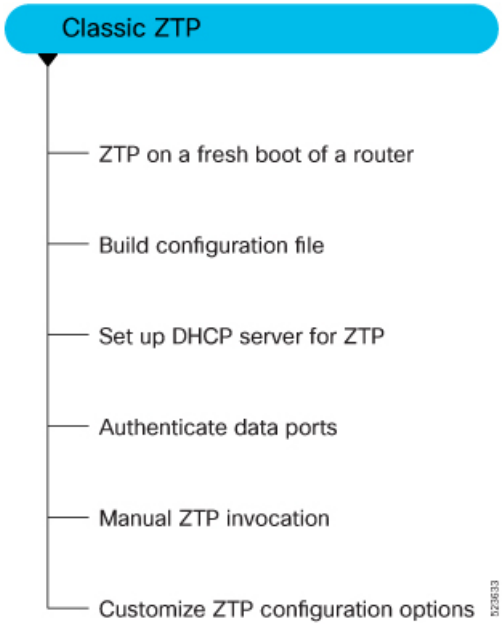
- Using ZTP to install Puppet

You can initiate ZTP in one of the following ways:

- **Fresh Boot**: Use this method for devices that has no pre-loaded configuration. See Getting Started with ZTP on a Fresh Boot of a Router. See Zero Touch Provisioning on a Fresh Boot of a Router, on page 105
- **Manual Invocation:** Use this method when you want to forcefully initiate ZTP on a fully configured device. See Invoke ZTP Manually, on page 106.

The following figure lists the tasks to perform to configure classic ZTP.

Figure 7: Workflow to Configure Classic ZTP



- Build your Configuration File, on page 92
- Authentication on Data Ports, on page 99
- Set Up DHCP Server, on page 101
- Customize ZTP Initialization File, on page 103
- Zero Touch Provisioning on a Fresh Boot of a Router, on page 105
- Invoke ZTP Manually, on page 106

Build your Configuration File

Based on the business need, you can use a configuration or script file to initiate the ZTP process.

Note When you use a USB flash drive as a source for ZTP, you cannot use the script file for provisioning. The script file is not supported in the USB fetcher. Fetcher defines which port the ZTP process should use to get the provisioning details as defined in the `ztp.ini` file.

The configuration file content starts with *!! IOS XR* and the script file content starts with *#! /bin/bash, #! /bin/sh or #!/usr/bin/python.*

Once you create the configuration file, apply it to the device using the *ztp_helper* function *xrapply*.

The following is the sample configuration file:

```
!! IOS XR
username root
group root-lr
password 0 lablab
!

hostname ios
alias exec al show alarms brief system active

interface HundredGigE 0/0/0/24
ipv4 address 10.10.10.55 255.255.255.0
no shutdown
!
```

Create User Script

This script or binary is executed in the IOS-XR Bash shell and can be used to interact with IOS-XR CLI to configure, verify the configured state and even run exec commands based on the workflow that the operator chooses.

Build your ZTP script with either shell and python. ZTP includes a set of CLI commands and a set of shell utilities that can be used within the user script.

Note We recommend that you do not execute the APIs on a router that is already provisioned. ZTP Utility APIs are designed to be executed from the ZTP script when you boot the router for the first time. The APIs perform additional operations to run the requested actions during the boot process and bring changes in the existing configuration before executing any action.

ZTP utility APIs have prerequisites that are executed in the ZTP workflow before running the ZTP utility APIs. These prerequisites help with running specific actions during the boot process and in making necessary configuration changes.

We recommend that you do not use ZTP utilities outside the scope of ZTP script. The APIs in this script use username as `ztp` or `ztp-user` in every action. The ZTP utility executed outside the scope of the ZTP script may fail as it is not executed from the ZTP workflow. This may modify the configurations on the device and affect other related operations. If the ZTP utility is executed outside the scope ZTP script, the logs display that the script is executed using username `ztp` or `ztp-user`, misleading that the script is executed from the workflow.

ZTP Shell Utilities

ZTP includes a set of shell utilities that can be sourced within the user script. `ztp_helper.sh` is a shell script that can be sourced by the user script. `ztp_helper.sh` provides simple utilities to access some XR functionalities. You can invoke the following bash functions:

- **xrcmd**—Used to run a single XR exec command:`xrcmd "show running"`
- **xrapply**—Applies the block of configuration, specified in a file:

```
cat >/tmp/config <<%%
!! XR config example
hostname node1-mgmt-via-xrapply
%%
xrapply /tmp/config
```

- **xrapply_with_reason**—Used to apply a block of XR configuration along with a reason for logging purpose:

```
cat >/tmp/config <<%%
!! XR config example
hostname node1-mgmt-via-xrapply
%%
xrapply_with_reason "this is a system upgrade" /tmp/config
```

- **xrapply_string**—Used to apply a block of XR configuration in one line:

```
xrapply_string "hostname foo\interface HundredGigE0/0/0/24\nipv4 address 1.2.3.44
255.255.255.0\n"
```

- **xrapply_string_with_reason**—Used to apply a block of XR configuration in one line along with a reason for logging purposes:

```
xrapply_string_with_reason ”system renamed again" "hostname venus\n interface
HundredGigE0/0/0/24\n
ipv4 address 172.30.0.144/24\n”
```

- **xrreplace**—Used to apply XR configuration replace in XR namespace via a file.

```
cat rtr.cfg <<%%
!! XR config example
hostname node1-mgmt-via-xrreplace
%%
xrreplace rtr.cfg
```

- **xrapply_with_extra_auth**—Used to apply XR configuration that requires authentication in XR namespace via a file. The **xrapply_with_extra_auth** API is used when configurations that require additional authentication to be applied such as alias, flex groups. This api internally performs authentication and authorization to gain additional privilege.

```
cat >/tmp/config <<%%
!! XR config example
alias exec alarms show alarms brief system active
alias exec version run cat /etc/show_version.txt
%%
xrapply_with_extra_auth >/tmp/config
```

- **xrreplace_with_extra_auth**—Used to apply XR configuration replace in XR namespace via a file. The **xrreplace_with_extra_auth** API is used when configurations that require additional authentication to be applied such as alias, flex groups. This api internally performs authentication and authorization to gain additional privilege.

```
cat >/tmp/config <<%%
!! XR config example
alias exec alarms show alarms brief system active
alias exec version run cat /etc/show_version.txt
```

```
%%
xrreplace_with_extra_auth >/tmp/config
```

API Implementation Behavior

Note The **xrcmd**,**xrapply**, and **xrreplace** APIs or utilities carry out a series of internal operations to execute specific actions. These operations, which are performed sequentially, include:

- User Creation—This operation involves generating a `ztp-user` (temporary user) before the execution of any other operations.
- Command Execution or Configuration Application—This operation encompasses executing a command, applying configurations using parser utilities, or applying the configuration through cfg-mgr.
- User Removal—This operation involves removing the `ztp-user` (temporary user) from the XR configuration.

In addition to these internal operations, the **xrapply_with_extra_auth** and **xrreplace_with_extra_auth** APIs performs an authentication process before applying configurations.

ZTP Helper Python Library

The ZTP python library defines a single Python class called `ZtpHelpers`. The helper script is located at `/pkg/bin/ztp_helper.sh`

ZtpHelpers Class Methods

Following are utility methods of the `ZtpHelpers` class:

- `init(self, syslog_server=None, syslog_port=None, syslog_file=None):`

```
__init__ constructor
:param syslog_server: IP address of reachable Syslog Server
       :param syslog_port: Port for the reachable syslog server
        :param syslog_file: Alternative or addon file for syslog
        :type syslog_server: str
        :type syslog_port: int
        :type syslog_file:str

         All parameters are optional. When nothing is specified during object creation,
 then all logs are sent to a log rotated file /tmp/ztp_python.log (max size of 1MB).
```

- `setns(cls, fd, nstype):`

```
Class Method for setting the network namespace
           :param cls: Reference to the class ZtpHelpers
           :param fd: incoming file descriptor
           :param nstype: namespace type for the sentns call
           :type nstype: int
                  0      Allow any type of namespace to be joined.
                  CLONE_NEWNET = 0x40000000 (since Linux 3.0)
                         fd must refer to a network namespace
```

- `get_netns_path(cls, nspath=None, nsname=None, nspid=None):`

```
Class Method to fetch the network namespace filepath
           associated with a PID or name
```

```
            :param cls: Reference to the class ZtpHelpers
            :param nspath: optional network namespace associated name
            :param nspid: optional network namespace associate PID
            :type nspath: str
            :type nspid: int
            :return: Return the complete file path
            :rtype:  str
```

- `toggle_debug(self, enable):`

  ```
  Enable/disable debug logging
            :param enable: Enable/Disable flag
            :type enable: int
  ```

- `set_vrf(self, vrfname=None):`

  ```
  Set the VRF (network namespace)
            :param vrfname: Network namespace name
                            corresponding to XR VRF
  ```

- `download_file(self, file_url, destination_folder):`

  ```
  Download a file from the specified URL
            :param file_url: Complete URL to download file
            :param destination_folder: Folder to store the
                                       downloaded file
            :type file_url: str
            :type destination_folder: str
            :return: Dictionary specifying download success/failure
                     Failure => { 'status' : 'error' }
                     Success => { 'status' : 'success',
                                  'filename' : 'Name of downloaded file',
                                  'folder' : 'Directory location of downloaded file'}
            :rtype: dict
  ```

- `setup_syslog(self):`

  Method to Correctly set sysloghandler in the correct VRF (network namespace) and point to a remote syslog Server or local file or default log-rotated log file.

- `xrcmd(self, cmd=None):`

  ```
  Issue an IOS-XR exec command and obtain the output
            :param cmd: Dictionary representing the XR exec cmd
                        and response to potential prompts
                        { 'exec_cmd': '', 'prompt_response': '' }
            :type cmd: dict
            :return: Return a dictionary with status and output
                     { 'status': 'error/success', 'output': '' }
            :rtype: dict
  ```

- `xrapply(self, filename=None, reason=None):`

  ```
  Apply Configuration to XR using a file
            :param file: Filepath for a config file
                         with the following structure:
                         !
                         XR config command
                         !
                         end

            :param reason: Reason for the config commit.
                           Will show up in the output of:
                           "show configuration commit list detail"
            :type filename: str
            :type reason: str
  ```

```
            :return: Dictionary specifying the effect of the config change
                     { 'status' : 'error/success', 'output': 'exec command based on
 status'}
                     In case of Error:  'output' = 'show configuration failed'
                     In case of Success: 'output' = 'show configuration commit changes
  last 1'
            :rtype: dict
```

- `xrapply_string(self, cmd=None, reason=None):`

```
 Apply Configuration to XR using  a single line string
            :param cmd: Single line string representing an XR config command
            :param reason: Reason for the config commit.
                           Will show up in the output of:
                           "show configuration commit list detail"
            :type cmd: str
            :type reason: str
            :return: Dictionary specifying the effect of the config change
                     { 'status' : 'error/success', 'output': 'exec command based on
 status'}
                     In case of Error:  'output' = 'show configuration failed'
                     In case of Success: 'output' = 'show configuration commit changes
  last 1'
            :rtype: dict
```

- `xrreplace(self, filename=None):`

```
 Replace XR Configuration using a file

            :param file: Filepath for a config file
                         with the following structure:

                         !
                         XR config commands
                         !
                         end
            :type filename: str
            :return: Dictionary specifying the effect of the config change
                     { 'status' : 'error/success', 'output': 'exec command based on
 status'}
                     In case of Error:  'output' = 'show configuration failed'
                     In case of Success: 'output' = 'show configuration commit changes
  last 1'
            :rtype: dict
```

API Implementation Behavior

Note The **xrcmd**,**xrapply**, and **xrreplace** APIs or utilities carry out a series of internal operations to execute specific actions. These operations, which are performed sequentially, include:

- User Creation—This operation involves generating a `ztp-user` (temporary user) before the execution of any other operations.
- Command Execution or Configuration Application—This operation encompasses executing a command, applying configurations using parser utilities, or applying the configuration through cfg-mgr.
- User Removal—This operation involves removing the `ztp-user` (temporary user) from the XR configuration.

Example

The following shows the sample script in phyton

```
[apple2:~]$ python sample_ztp_script.py

###### Debugs enabled ######


###### Change context to user specified VRF ######


###### Using Child class method, setting the root user ######

2016-12-17 04:23:24,091 - DebugZTPLogger - DEBUG - Config File content to be applied  !
                    username netops
                    group root-lr
                    group cisco-support
                    secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1
                    !
                    end
2016-12-17 04:23:28,546 - DebugZTPLogger - DEBUG - Received exec command request: "show
configuration commit changes last 1"
2016-12-17 04:23:28,546 - DebugZTPLogger - DEBUG - Response to any expected prompt ""
Building configuration...
2016-12-17 04:23:29,329 - DebugZTPLogger - DEBUG - Exec command output is ['!! IOS XR
Configuration version = 6.2.1.21I', 'username netops', 'group root-lr', 'group cisco-support',
 'secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1', '!', 'end']
2016-12-17 04:23:29,330 - DebugZTPLogger - DEBUG - Config apply through file successful,
last change = ['!! IOS XR Configuration version = 6.2.1.21I', 'username netops', 'group
root-lr', 'group cisco-support', 'secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1', '!', 'end']


###### Debugs Disabled ######


###### Executing a show command ######

Building configuration...
{'output': ['!! IOS XR Configuration version = 6.2.1.21I',
            '!! Last configuration change at Sat Dec 17 04:23:25 2016 by UNKNOWN',
            '!',
            'hostname customer2',
            'username root',
            'group root-lr',
            'group cisco-support',
            'secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1',
            '!',
            'username noc',
            'group root-lr',
            'group cisco-support',
            'secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1',
            '!',
            'username netops',
            'group root-lr',
            'group cisco-support',
            'secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1',
            '!',
            'username netops2',
            'group root-lr',
            'group cisco-support',
            'secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1',
            '!',
            'username netops3',
            'group root-lr',
```

```
            'group cisco-support',
            'secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1',
            '!',
            'cdp',
            'service cli interactive disable',
            'interface MgmtEth0/RP0/CPU0/0',
            'ipv4 address 11.11.11.59 255.255.255.0',
            '!',
            'interface TenGigE0/0/0/0/24',
            'shutdown',
            '!',
            'interface TenGigE0/0/0/0/25',
            'shutdown',
            '!',

            'router static',
            'address-family ipv4 unicast',
            '0.0.0.0/0 11.11.11.2',
            '!',
            '!',
            'end'],
 'status': 'success'}

###### Apply valid configuration using a file ######

Building configuration...
{'status': 'success', 'output': ['!! IOS XR Configuration version = 6.2.1.21I', 'hostname
customer', 'cdp', 'end']}

###### Apply valid configuration using a string ######

Building configuration...
{'output': ['!! IOS XR Configuration version = 6.2.1.21I',
            'hostname customer2',
            'end'],
 'status': 'success'}

###### Apply invalid configuration using a string ######

{'output': ['!! SYNTAX/AUTHORIZATION ERRORS: This configuration failed due to',
            '!! one or more of the following reasons:',
            '!!  - the entered commands do not exist,',
            '!!  - the entered commands have errors in their syntax,',
            '!!  - the software packages containing the commands are not active,',
```

For information on helper APIs, see https://github.com/ios-xr/iosxr-ztp-python#iosxr-ztp-python.

Authentication on Data Ports

On fresh boot, ZTP process is initiated from management ports and may switch to data ports. To validate the connection with DHCP server, authentication is performed on data ports through DHCP option 43 for IPv4 and option 17 for IPv6. These DHCP options are defined in option space and are included within **dhcpd.conf** and **dhcpd6.conf** configuration files. You must provide following parameters for authentication while defining option space:

- Authentication code—The authentication code is either 0 or 1; where 0 indicates that authentication is not required, and 1 indicates that MD5 checksum is required.

Note If the option 43 for IPv4, and option 17 for IPv6 is disabled, the authentication fails.

- Client identifier—The client identifier must be 'xr-config'.
- MD5 checksum—This is chassis serial number. It can be obtained using **echo -n $SERIALNUMBER | md5sum | awk '{print $1}'** .

Here is the sample **dhcpd.conf** configuration. In the example below, the option space called **VendorInfo** is defined with three parameters for authentication:

```
class "vendor-classes" {
   match option vendor-class-identifier;
}

option space VendorInfo;
option VendorInfo.clientId code 1 = string;
option VendorInfo.authCode code 2 = unsigned integer 8;
option VendorInfo.md5sum code 3 = string
option vendor-specific code 43 = encapsulate VendorInfo;
subnet 10.65.2.0 netmask 255.255.255.0 {
  option subnet-mask 255.255.255.0;
  option routers 10.65.2.1;
  range 10.65.2.1 10.65.2.200;
}
host cisco-mgmt {
   hardware ethernet 00:50:60:45:67:01;
   fixed-address 10.65.2.39;
   vendor-option-space VendorInfo;
   option VendorInfo.clientId  "xr-config";
   option VendorInfo.authCode 1;
   option VendorInfo.md5sum "aedf5c457c36390c664f5942ac1ae3829";
   option bootfile-name "http://10.65.2.1:8800/admin-cmd.sh";
}
```

Here is the sample **dhcpd6.conf** configuration file. In the example below, the option space called **VendorInfo** is defined that has code width 2 and length width 2 (as per dhcp standard for IPv6) with three parameters for authentication:

```
log-facility local7;
option dhcp6.name-servers 2001:1451:c632:1::1;
option dhcp6.domain-search "cisco.com";
dhcpv6-lease-file-name "/var/lib/dhcpd/dhcpd6.leases";
option dhcp6.info-refresh-time 21600;
option dhcp6.bootfile-url code 59 = string;
option dhcp6.user-class code 15 = string;
option space CISCO-XR-CONFIG code width 2 length width 2;
option CISCO-XR-CONFIG.client-identifier code 1 = string;
option CISCO-XR-CONFIG.authCode code 2 = integer 8;
option CISCO-XR-CONFIG.md5sum code 3 = string;
option vsio.CISCO-XR-CONFIG code 9 = encapsulate CISCO-XR-CONFIG;
subnet6 2001:1451:c632:1::/64{
 range6 2001:1451:c632:1::2 2001:1451:c632:1::9;
 option CISCO-XR-CONFIG.client-identifier "xr-config";
 option CISCO-XR-CONFIG.authCode 1;
 #valid md5
 option CISCO-XR-CONFIG.md5sum "90fd845ac82c77f834d57a034658d0f0";
 if option dhcp6.user-class = 00:04:69:50:58:45 {
```

```
  option dhcp6.bootfile-url "http://[2001:1851:c632:1::1]/cisco-2/image.iso";
 }
 else {
   #option dhcp6.bootfile-url "http://[2001:1851:c632:1::1]/cisco-2/cisco-mini-x.iso.sh";
   option dhcp6.bootfile-url "http://[2001:1851:c632:1::1]/cisco-2/ztp.cfg";
  }
}
```

Set Up DHCP Server

For ZTP to operate a valid IPv4 or IPv6 address is required and the DHCP server must send a pointer to the configuration script.

The DHCP request from the router has the following DHCP options to identify itself:

- **Option 60**: “vendor-class-identifier” : Used to Identify the following four elements:
  - The type of client: For example, PXEClient
  - The architecture of The system (Arch): For example: 00009 Identify an EFI system using a x86-64 CPU
  - The Universal Network Driver Interface (UNDI):

    For example 003010 (first 3 octets identify the major version and last 3 octets identify the minor version)
  - The Product Identifier (PID):
- **Option 61**: “dhcp-client-identifier” : Used to identify the Serial Number of the device.
- **Option 66** : Used to request the TFTP server name.
- **Option 67**: Used request the TFTP filename.
- **Option 97**: “uuid” : Used to identify the Universally Unique Identifier a 128-bit value (not usable at this time)

Example

The following DHCP request sample provides a fixed IP address and a configuration file with the mac address of the management interface.

```
host cisco-rp0 {
   hardware ethernet e4:c7:22:be:10:ba;
   fixed-address 172.30.12.54;
   filename "http://172.30.0.22/configs/cisco-1.config";
}
```

The following DHCP request sample provides a fixed IP address and a configuration file with the mac address of the management interface along with capability to re-image the system using iPXE ("xr-config" option):

```
host cisco-rp0 {
   hardware ethernet e4:c7:22:be:10:ba;
   fixed-address 172.30.12.54;
   if exists user-class and option user-class = "iPXE" {
      filename = "http://172.30.0.22/boot.ipxe";
   } elsif exists user-class and option user-class =  "xr-config" {
      filename = "http://172.30.0.22/scripts/cisco-rp0_ztp.sh";
```

```
   }
}
```

DHCP server identifies the device and responds with either an IOS-XR configuration file or a ZTP script as the filename option.

The DHCP server responds with the following DHCP options:

- DHCPv4 using BOOTP filename to supply script/config location.
- DHCPv4 using Option 67 (bootfile-name) to supply script/config location.
- DHCPv6 using Option 15: If you have configured this option for the server to identify ztp requests, ensure that you update the server configuration, for Linux or ISC servers. Sample server-side configuration required to check user-class for ZTP is shown in the following example:

  ```
  if exists dhcp6.user-class and (substring(option dhcp6.user-class, 0, 9) = "xr-config"
   or substring(option dhcp6.user-class, 2, 9) = "xr-config"){
       #
  }
  ```

- DHCPv6 using Option 59 (OPT_BOOTFILE_URL) to supply script/config location

The following sample shows the DHCP response with bootfile-name (option 67):

```
option space cisco-vendor-id-vendor-class code width 1 length width 1;
option vendor-class.cisco-vendor-id-vendor-class code 9 = {string};

######### Network 11.11.11.0/24 #################
shared-network 11-11-11-0 {

####### Pools ###############
  subnet 11.11.11.0 netmask 255.255.255.0 {
    option subnet-mask 255.255.255.0;
 option broadcast-address 11.11.11.255;
 option routers 11.11.11.2;
 option domain-name-servers 11.11.11.2;
 option domain-name "cisco.local";
 # DDNS statements
   ddns-domainname "cisco.local.";
 # use this domain name to update A RR (forward map)
   ddns-rev-domainname "in-addr.arpa.";
   # use this domain name to update PTR RR (reverse map)

    }

######## Matching Classes ##########

  class "cisco" {
       match if (substring(option dhcp-client-identifier,0,11) = "FGE194714QS");
  }


  pool {
      allow members of "cisco";
      range 11.11.11.47 11.11.11.50;
      next-server 11.11.11.2;

      if exists user-class and option user-class = "iPXE" {
         filename="http://11.11.11.2:9090/cisco-mini-x-6.2.25.10I.iso";
      }


      if exists user-class and option user-class =  "xr-config"
```

```
        {
           if (substring(option vendor-class.cisco-vendor-id-vendor-class,19,99)="cisco")
           {
            option bootfile-name "http://11.11.11.2:9090/scripts/exhaustive_ztp_script.py";

           }
         }


      ddns-hostname "cisco-local";
      option routers 11.11.11.2;
  }
}
```

Important In Cisco IOS XR Release 7.3.1 and earlier, the system accepts the device sending **user-class = "exr-config"**; however starting Cisco IOS XR Release 7.3.2 and later, you must use only **user-class = "xr-config"**.

In Cisco IOS XR Release 7.3.2 and later, use:

```
host cisco-rp0 {
   hardware ethernet e4:c7:22:be:10:ba;
   fixed-address 172.30.12.54;
   if exists user-class and option user-class = "iPXE" {
      filename = "http://172.30.0.22/boot.ipxe";
   } elsif exists user-class and option user-class = "xr-config" {
      filename = "http://172.30.0.22/scripts/cisco-rp0_ztp.sh";
   }
}
```

Also, when upgrading from any release that is Cisco IOS XR Release 7.3.1 or earlier to Cisco IOS XR Release 7.3.2 or later release, use the following:

```
host cisco-rp0 {
   hardware ethernet e4:c7:22:be:10:ba;
   fixed-address 172.30.12.54;
   if exists user-class and option user-class = "iPXE" {
      filename = "http://172.30.0.22/boot.ipxe";
   } elsif exists user-class and option user-class = "exr-config" {
      filename = "http://172.30.0.22/scripts/cisco-rp0_ztp.sh";
   }
}
```

Customize ZTP Initialization File

You can customize the following ZTP configurable options in the *ztp.ini* file:

- `ZTP`: You can enable or disable ZTP at boot using CLI or by editing the *ztp.ini* file.
- `Retry`: Set the ZTP DHCP retry mechanism: The available values are infinite and once.
- `Fetcher Priority`: Fetcher defines which port ZTP should use to get the provisioning details. By default, each port has a fetcher priority defined in the *ztp.ini* file. You can modify the default priority of the fetcher. Allowed range is 0–10.

Note Lower the number higher the priority. The value 0 has the highest priority and 10 has the lowest priority.

In the following example, the Mgmt4 port has the highest priority:

```
[Fetcher Priority]
Mgmt4: 0
Mgmt6: 1
DPort4: 2
DPort6: 3
```

- `progress_bar`: Enable progress bar on the console. By default, the progress bar is disabled. To enable the progress bar, add the following entry in the `ztp.ini` file.

```
[Options]
progress_bar: True
```

By default, the `ztp.ini` file is located in the `/pkg/etc/` location. To modify the ZTP configurable options, make a copy of the file in the `/disk0:/ztp/` directory and then edit the `ztp.ini` file.

To reset to the default options, delete the ztp.ini file in the `/disk0:/ztp/` directory.

Note Do not edit or delete the `ztp.ini` file in the `/pkg/etc/` location to avoid issues during installation.

The following example shows the sample of the `ztp.ini` file:

```
[Startup]
start: True
retry_forever: True

[Fetcher Priority]
Mgmt4:  1
Mgmt6:  2
DPort4: 3
DPort6: 4
```

Enable ZTP Using CLI

If you want to enable ZTP using CLI, use the **ztp enable** command.

Configuration example

```
Router#ztp enable
Fri Jul 12 16:09:02.154 UTC
Enable ZTP? [confirm] [y/n] :y
ZTP Enabled.
```

Disable ZTP Using CLI

If you want to disable ZTP using CLI, use the **ztp disable** command.

Configuration example

```
Router#ztp disable
Fri Jul 12 16:07:18.491 UTC
Disable ZTP? [confirm] [y/n] :y
```

```
ZTP Disabled.
Run ZTP enable to run ZTP again.
```

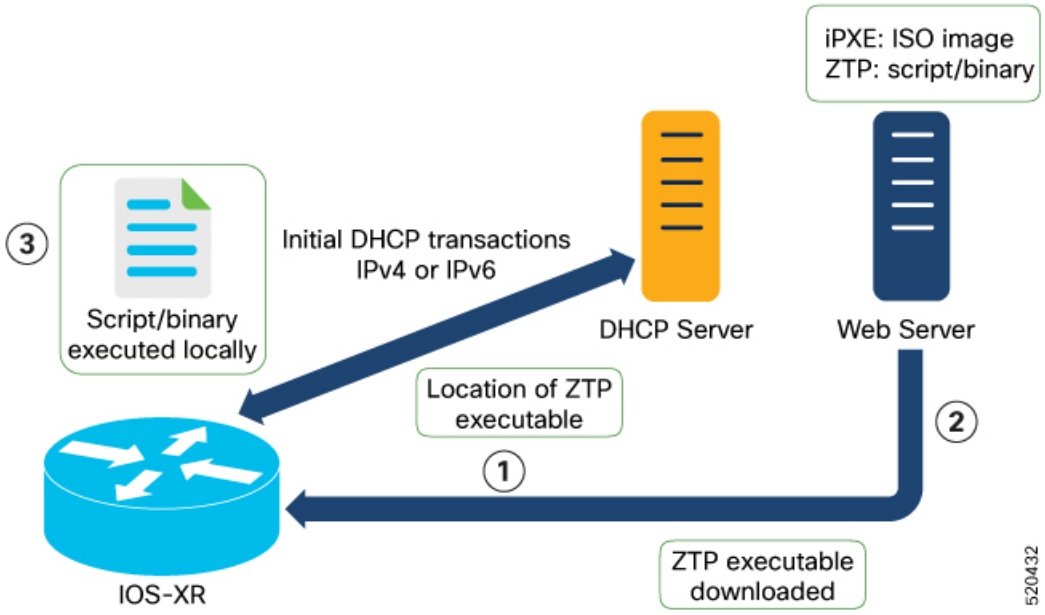
Zero Touch Provisioning on a Fresh Boot of a Router

When you boot the device, the ZTP process initiates automatically if the device does not have a prior configuration.

Fresh Boot Using DHCP

When you boot the device, the ZTP process initiates automatically if the device does not have a prior configuration. During the process, the router receives the details of the configuration file from the DHCP server.

This image depicts the high-level work flow of the ZTP process:



The ZTP process initiates when you boot the network-device with an IOS-XR image. The process starts only on the device that doesn't have a prior configuration.

Here is the high-level work flow of the ZTP process for the Fresh boot:

1. ZTP sends DHCP request to fetch the ZTP configuration file or user script. To help the Bootstrap server uniquely identify the device, ZTP sends below DHCP option
   - DHCP(v4/v6) client-id=Serial Number
   - DHCPv4 option 124: Vendor, Platform, Serial-Number
   - DHCPv6 option 16: Vendor, Platform, Serial-Number

   The following is the default sequential flow of the ZTP process:

- ZTP sends IPv4 DHCP request first on all the management port. In case there is a failure, then ZTP sends IPv6 DHCP request on all the management port.
- ZTP sends IPv4 DHCP request first on all the data port. In case there is a failure, then ZTP sends IPv6 DHCP request on all the data port.

  The default sequential flow is defined in configuration file and you can modify the sequence using the configuration file.

2. DHCP server identifies the device and responds with DHCP response using one of the following options:

   DHCP server should be configured to respond with the DHCP options.

   - DHCPv4 using BOOTP filename to supply script/config location.
   - DHCPv4 using Option 67 (bootfile-name) to supply script/config location.
   - DHCPv6 using Option 59 (OPT_BOOTFILE_URL) to supply script/config location
3. The network device downloads the file from the web server using the URI location that is provided in the DHCP response.
4. The device receives a configuration file or script file from the HTTP server.

Note
- If the downloaded file content starts with !! IOS XR it is considered as a configuration file.
- If the downloaded file content starts with #! /bin/bash, #! /bin/sh or #!/usr/bin/python it is considered as a script file.

5. The device applies the configuration file or executes the script or binary in the default bash shell.
6. The Network device is now up and running.

Invoke ZTP Manually

You can invoke Zero Touch Provisioning (ZTP) manually through the Command Line Interface. This method is Ideal for verifying the ZTP configuration without a reboot. This manual approach helps you to provision the router in stages. To invoke ZTP on an interface (data ports or management port), you don't have to bring up and configure the interface first.

Even when the interface is down, you can run the `ztp initiate` command, and the ZTP script will bring it up and invoke `dhclient`. Hence, ZTP can run on all interfaces irrespective of their availability.

Use the following commands to manually invoke the ZTP commands and to force ZTP to run on all interfaces:

- **ztp initiate** — Invokes a new ZTP DHCP session. Logs can be found in **/disk0:/ztp/ztp.log**.

  Configuration Example:

```
Router#ztp initiate debug verbose interface HundredGigE 0/0/0/24
Invoke ZTP? (this may change your configuration) [confirm] [y/n] :
```

- **ztp terminate** —Terminates any ZTP session in progress.

Configuration Example:

```
Router #ztp terminate verbose
Mon Oct 10 16:52:38.507 UTC
Terminate ZTP? (this may leave your system in a partially configured state) [confirm]
[y/n] :y
ZTP terminated
```

- **ztp enable** —Enables the ZTP at boot.

  Configuration Example:

  ```
  Router#ztp enable
  Fri Jul 12 16:09:02.154 UTC
  Enable ZTP? [confirm] [y/n] :y
  ZTP Enabled.
  ```

- **ztp disable** —Disables the ZTP at boot.

  Configuration Example:

  ```
  Router#ztp disable
  Fri Jul 12 16:07:18.491 UTC
  Disable ZTP? [confirm] [y/n] :y
  ZTP Disabled.
  Run ZTP enable to run ZTP again.
  ```

- **ztp clean** —Removes only the ZTP state files.

  Configuration Example:

  ```
  Router#ztp clean verbose
  Mon Oct 10 17:03:43.581 UTC
  Remove all ZTP temporary files and logs? [confirm] [y/n] :y
  All ZTP files have been removed.
  If you now wish ZTP to run again from boot, do 'conf t/commit replace' followed by
  reload.
  ```

  The log file ztp.log is saved in **/var/log** folder, and a copy of log file is available at **/disk0:/ztp/ztp.log** location using a soft link. However, executing **ztp clean** clears files saved on disk and not on **/var/log** folder where current ZTP logs are saved. In order to have a log from current ZTP run, you must manually clear the ZTP log file from **/var/log/** folder.

Configuration

This task shows the most common use case of manual ZTP invocation: invoke ZTP.

1. Invoke DHCP sessions on all data ports which are up or could be brought up. ZTP runs in the background. Use show logging or look at /disk0:/ztp/ztp.log to check progress.

Configuration Example:

```
Router# ztp initiate dataport
```