

Cisco IOS XR Setup and Upgrade Guide for Cisco 8000 Series Routers

First Published: 2023-03-01

Last Modified: 2026-09-11

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

© 2023–2026 Cisco Systems, Inc. All rights reserved.



CONTENTS

CHAPTER 1 **Key Concepts** 1

Key Terms and Concepts 1

Types of Releases 5

Files in Cisco Software Download Page 5

Command Modes 6

CHAPTER 2 **Workflow to Setup and Upgrade the Router** 7

Setup and Upgrade Workflow 7

CHAPTER 3 **Setup the Router** 11

Prerequisites to Setup Router 11

Connect Console Port to Terminal 11

Install Remote Management Protocols 13

Setup the Router 13

Boot the Router 14

Configure IP Address and Subnet Mask 14

Synchronize Router Clock with NTP Server 16

Verify the Software and Hardware Status 17

Verify Software Version 18

Verify Hardware Modules 18

Verify Interface Status 21

Verify Node Status 22

Complete Post-setup Tasks 24

Create User Profile 25

Create User Groups 26

CHAPTER 4

Upgrade the Router 29

Plan the Software Upgrade 29

Supported upgrade and downgrade releases 30

Prerequisites 32

Supported software upgrade from running version 32

Supported releases to upgrade software from current version to target version 33

Show commands for software upgrade and downgrade 33

Backup Current Configuration 34

Check FPD Version 35

Upgrading FPDs Using Yang Data Models 37

Check System Stability 37

Mitigate Traffic Loss During Upgrade 38

Optional packages 39

Limitations of optional packages 40

Commands and YANG data models supported using optional RPMs 40

Obtain Install Files 41

Standard ISO and RPMs 41

Golden ISO images 41

Create Repository to Access Install Files 43

Create Remote Repository 44

Create Local Repository on the Router 45

Disk encryption 46

How disk encryption works 47

Restrictions for disk encryption 48

Guidelines for disk encryption 48

Transactionality for release upgrades and downgrades 49

Prerequistes for transactionality 50

Guidelines for transactionality 50

Restrictions for transactionality 50

Upgrade the Software 51

Upgrade Router Using CLI Commands 51

Install IOS XR Image 52

Upgrade Router Using YANG Data Models 57

Access Install-related Data Models 57
Use Manageability Agent to Connect to Router 58
Generate RPC Messages to Install IOS XR Image 59
Upgrade QDD Optical Modules 60
HTTPS copy and install operation requirements 61
Verify the Software Upgrade 61
Check System Stability 62
View supported features and capabilities 63
Optics firmware upgrades 67
Upgrade optics firmware 68
Deprecated commands 70

CHAPTER 5 **Deploy Router Using Bootz** 71
Supported Bootz versions 72
Components used in the Bootz Process 72
Onboard Devices Using Bootz Workflow 73
Obtain Ownership Voucher 74
Build bootstrapping data 75
Bootz workflow commit process for hw-module commands 76
Provision Bootz Using DHCP Server 76
Bootz Workflow for Standby RP 82
Overview 84
Prerequisites 84
Restrictions 84
Use Cases 84
How the Router Obtains and Processes the OV Information 85
BootstrapStream RPC for Bootz process 86
Limitations of BootstrapStream RPC 88
How BootstrapStream RPC operations work 88
Immutable bootstrap configurations 89
Limitations of immutable bootstrap configuration 91
Verify immutable bootstrap configuration on the router 91
Bootz dynamic configuration 92
Guidelines for Bootz dynamic configuration 94

How Bootz dynamic configuration works 94
Verify Bootz dynamic configuration 96

CHAPTER 6 **Deploy Router Using Secure ZTP 99**
Obtain Ownership Voucher 102
Build Bootstrapping Data 103
Secure ZTP Options 106
Provision secure ZTP using a USB drive 106
Provision Secure ZTP Using DHCP Server 108

CHAPTER 7 **Deploy Router Using Classic ZTP 115**
Deploy Router Using Classic ZTP 117
Build Configuration File 118
Configuration Data Using JSON-encoded YANG Information 128
Limitations of temporary ZTP files 134
Authenticate Data Ports 135
Setup DHCP Server 138
Customize ZTP Initialization File 141
ZTP file download behavior with libcurl 144
Zero Touch Provisioning on a Fresh Boot of a Router 144
Fresh boot using removable storage device 145
Fresh Boot Using DHCP 147
Manual Invocation of ZTP 148

CHAPTER 8 **Additional Install Operations 151**
Install Additional RPMs and Bug Fixes 152
Option 1: Install RPMs Using Command Line Interface 153
Option 2: Install RPMs Using YANG Data Model 155
Downgrade Software Version 155
Remove a bug fix from Cisco IOS XR software 157
Roll back software to a previously saved installation point 158
Roll back Cisco IOS XR software using a rollback ID 159
Rollback from SONiC to Cisco IOS XR OS 161
Stream Telemetry Data for Install Operations 163

CHAPTER 9 **Troubleshoot Router Setup and Upgrade** 165

Recover router from boot failure 166

Boot the router using USB drive 166

Boot the router using iPXE 168

Boot the router using install replace reimage 171

Recover password 172

Rectify insufficient disk space when installing software 174

Recover frozen console prompt 176

CHAPTER 10 **Install Owner and Partner RPMs using IOS XR Install Infrastructure** 179

Owner and partner RPMs 179

Limitations and Guidelines 182

Security posture transitions 183

Installing owner and partner RPMs 183

Install unsigned owner RPMs 184

Install signed owner RPMs with or without signature verification 185

Install signed owner RPMs using GISO 185

Install signed owner RPMs using USB or iPXE boot 186

Two-step upgrade process for installing owner or partner RPMs 187

Third-party unsigned RPMs 187

Guidelines for installing unsigned third-party RPMs 188

Install third-party unsigned RPMs 188

Troubleshooting Installation Failures 189

CHAPTER 11 **Upgrading Field-Programmable Device** 191

Overview of FPD Image Upgrade 191

Restrictions for FPD Upgrade 191

Types of FPD Upgrade Service 192

Manual FPD upgrade 192

Automatic FPD Upgrade 193

How to Upgrade FPD Images 194

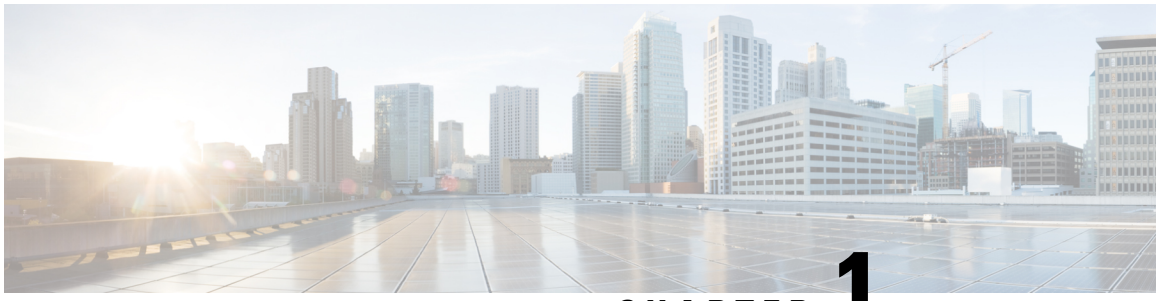
Automatic Line Card Reload on FPD Upgrade 200

Restrictions for Automatic Line Card Reload on FPD Upgrade 200

Configure Automatic Line Card Reload on FPD Upgrade 200
Power Module Upgrades 200
Manual Power Module FPD Upgrade 200
Upgrading FPD for PSU 202
Automatic FPD Upgrade for PSU 203
Exclude the Default PSU Upgrade from the Automatic FPD Upgrade 204
Auto upgrade support for SC/MPA 205

CHAPTER 12 Release-specific Caveats and Workarounds 207
Release 25.3.1 207
Release 25.2.1 209
Release 7.10.1 211
Release 7.9.1 213
Release 7.7.2 214
Release 7.7.1 214
Release 7.5.2, Release 7.5.3 215
Release 7.5.1, Release 7.3.2 216

CHAPTER 13 Setup and Upgrade Commands 217
Action Commands 217
Show Commands 217



CHAPTER 1

Key Concepts

Use this information to understand the key terms, concepts, types of releases relevant to setting up and upgrading Cisco IOS XR software on Cisco 8000 series routers.

This section contains the following topics:

- Key Terms and Concepts, on page 1
- Types of Releases, on page 5
- Files in Cisco Software Download Page, on page 5
- Command Modes, on page 6

Key Terms and Concepts

Applicable Variants for Cisco 8000 Series Routers

The Cisco 8000 series routers run on IOS XR software with XR7 architecture. The procedures for setting up and upgrading the software are applicable for these variants of the series:

- Cisco 8201
- Cisco 8202
- Cisco 8808
- Cisco 8812
- Cisco 8818

Setup

When the router with the pre-installed software is powered ON for the first time, the pre-installed version of the IOS XR software starts functioning automatically. You set up and configure the router for network capabilities.

Upgrade

When a new version of the IOS XR software is available, you may choose to upgrade to that version.

Upgrade Methods

After your upgrade, you apply the changes by using one of two methods:

- Process restart – the system is upgraded while it's in service by restarting only those processes that are affected by the upgrade. This applies to Software Maintenance Upgrades (SMUs).

  However, some SMUs can be applied during system reload as well.

- Drain and reboot – traffic is drained before the upgrade and changes are applied by reloading the router with the updated version.

Packages and Red Hat Package Manager (RPM)

Cisco uses the Red Hat Package Manager (RPM) package management system to package required and optional files for installing and upgrading the IOS XR software.

Base Image and Optional Packages

The .iso is the bare minimum software image that is required to run IOS XR on the router. Additional IOS XR packages are optional and are needed depending on the router configuration and required features. If you wish to add features like multicast, manageability, BNG, you must install the appropriate optional package.

Customizable ISO or Golden ISO (GISO)

Golden ISO (GISO) is a customized ISO image that is built to contain preferable packages to suit diverse installation requirements. GISO can be customized to include a standard base image with the basic functional components, additional RPMs, bug fixes, and configuration files based on your requirement.

Active and Committed Packages

An *active package* is the software version on the router after a version upgrade. If the upgraded version is to be retained after reloads, you need to *commit* the changes. After committing the changes, the active packages would then match the list of committed packges retained till the next applicable install upgrade or downgrade operation, or until the install package-based operations are carried out.

Operation

An operation is the sum of all work carried out to fulfill a user’s request provided through CLIs or RPCs.

The internal work performed to complete an operation is regarded to be at different levels of operation: transactions, atomic software changes, and packaging operations.

Each operation is assigned an Operation ID, which is a function of one of these:

Table 1: Operation IDs

Operation ID	Function
1	Transaction ID
1.1	Transaction ID and the atomic change ID if there is one
1.1.1	Transaction ID, the atomic change ID, and packaging operation ID if there is one

For example, these are operations with their Operation IDs carried out through these commands.

- install package add xr-bgp – **1.1.1** - This is starting the **first** transaction, the **first** atomic operation and the **first** packaging operation.
- install package remove xr-bgp – 1.1.**2** - This is starting a **second** packaging operation, within the first atomic operation in the first transaction.

Transactions

Transactions are the highest level of operation. Starting a transaction marks the start of an overall operation. To maintain the software changes carried out during a transaction, you must commit the transaction. If the system reloads during an install transaction, the running software is reverted to its previous state before the transaction was started. Within a transaction, multiple atomic software changes can be performed.

Atomic Software Changes

All atomic changes occur within a transaction. During an atomic software change, any changes to install IOS XR software are not visible to the system. The changes become visible when the atomic change is applied. Within an atomic operation, multiple packaging operations can be performed.

An atomic operation occurs in its entirety, or does not occur it all. During an upgrade, there is a switchover from the old software to the new in a single step.

Packaging Operations

Packaging operations are actions performed to change the packages that are installed on the system. Every packaging operation is contained within an atomic change. Atomic changes may contain multiple packaging operations. Examples of packaging operations are upgrade, downgrade, replace, add, or remove packages.

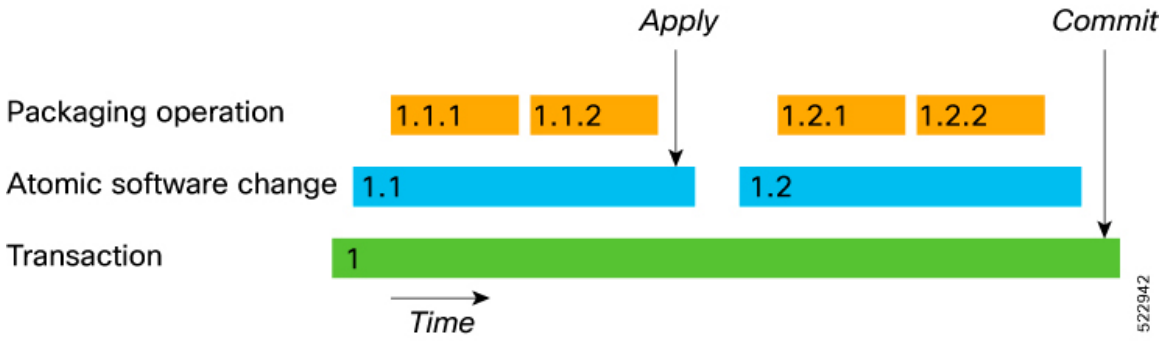
Apply and Commit

One successfully completed operation that modifies the software involves three phases.

- The internal **execution** of the packaging operation that changes the packages that are installed on the system.
- The **Apply** phase that completes an atomic software change and makes the software change visible to the system.
- The **Commit** phase that ends a transaction and ensures that all software changes continue to be present when the router is reloaded.

The following figure shows how the key operations concepts fit together in install and upgrade operations:

Figure 1: Transaction, Atomic Software Change, and Packaging Operation



Example:

- install package add xr-bgp – **1.1.1** This is starting the **first** transaction, the **first** atomic operation and the **first** packaging operation.
- install package remove xr-bgp – 1.1.**2** This is starting a **second** packaging operation, within the first atomic operation in the first transaction.
- install apply - 1.1 This is applying the first atomic operation in the first transaction
- install package add xr-bgp – 1.**2.1**This is starting the **first** packaging operation and the **second** atomic change in the first transaction.
- install package remove xr-bgp – 1.2.**2** This is starting the **second** packaging operation in the second atomic change in the first transaction.
- install apply – 1.2 This is applying the first atomic change in the first transaction.
- install commit - 1This is committing the first transaction
- install package add xr-bgp – **2.1.1** This is starting the **second** transaction, the **first** atomic operation within that transaction and the **first** packaging operation within that atomic operation

Synchronous Action

An asynchronous action allows you to gain access to the prompt and perform another parallel task as the install or upgrade operation continues to its completion.

When installing or upgrading, you can request a synchronous action. Specify the keyword *synchronous* in the install commands, and the prompt is returned only when the request has completed, the Ctrl + C keys are pressed, or a reload occurs.

When the synchronous action is in effect, the user is updated with the status of the request whenever it changes. Pressing Ctrl + C keys during a synchronous action request returns the prompt to the user but does not halt the install or upgrade operation.

Types of Releases

Cisco IOS XR software model has three types of software releases. The software images are available for download at the Cisco Software Download page.

Feature Release

A Feature Release (FR) contains new features and support for new hardware. Feature releases have the X.X.1 (dot one) designation. For example, releases 7.1.1 and 7.5.1 are feature releases. The list of features added to a feature release is provided in the Release notes along with the installation instructions and dependencies.

Maintenance Release

A maintenance release is the primary mechanism to deliver groups of critical bug fixes to the software feature releases.

SMUs Release

A Software Maintenance Unit (SMU) is a fix that is provided until the End of Maintenance (EoM) of the release. The fix is also committed into the next shipping release. SMUs are posted under `Cisco IOS XR Software Maintenance Upgrade` on the Cisco Software Download page. Each SMU is customized for a specific software release.

For more information on release numbering, types of releases, and their timelines, see Software Lifecycle Support Statement - IOS XR.

Files in Cisco Software Download Page

The following table describes the files available for download from the Cisco Software Download page for each variant of the Cisco 8000 series routers:

Table 2: IOS XR Software Installation Files in Cisco Software Download Page

Package File	Example	Description
8000-x64-<*rel. no.*>.iso	8000-x64-7.9.1.iso	Bootable ISO Image of the Operating System required to run IOS XR on a device for basic operations
8000-usb_boot-<*rel. no.*>.zip	8000-x64-usb-7.9.1.zip	USB Boot image of the Operating System
8000-optional-rpms. <*rel. no.*>.tar	8000-optional-rpms.7.9.1.tar	Optional RPMs that provide additional functionality
8000-k9sec-rpms. <*rel. no.*>.tar	8000-k9sec-rpms.7.9.1.tar	Security package that includes software that uses encryption (e.g. SSH) and has export controls for downloads of this software

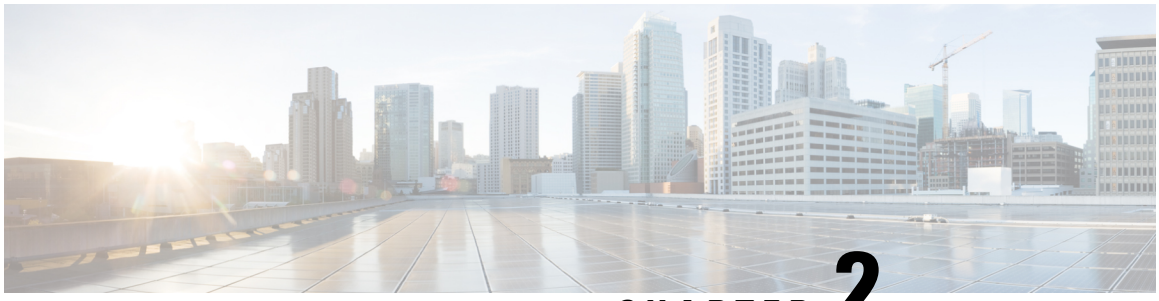
Package File	Example	Description
8000-<*rel. no.*><*bug-ID*>.tar)	8000-7.9.1.CSCvy99756.tar	Optional or recommended SMUs

Command Modes

The router runs on virtualized Cisco IOS XR software. Therefore, the CLI commands must be executed on virtual machines, namely the XR LXC and the System Admin LXC.

The command modes are applicable to the Cisco Series Routers. This table lists the command modes for the LXCs.

Command Mode	Description
XR EXEC mode (XR LXC execution mode)	Run commands on the XR LXC to display the operational state of the router. Example: `RP/0/RP0/CPU0:router#`
XR Config mode (XR LXC configuration mode)	Perform security, routing, and other XR feature configurations on the XR LXC. Example: `RP/0/RP0/CPU0:router#`**`configure`** `RP/0/RP0/CPU0:router(config)#`



CHAPTER 2

Workflow to Setup and Upgrade the Router

The setup and upgrade process depends on several factors. Each process is composed of a series of tasks, forming a linear progression that guides you through completing the tasks. Although there may be differences in certain tasks depending on a specific scenario, some tasks are common across multiple journeys.

The processes are outlined in the following topic:

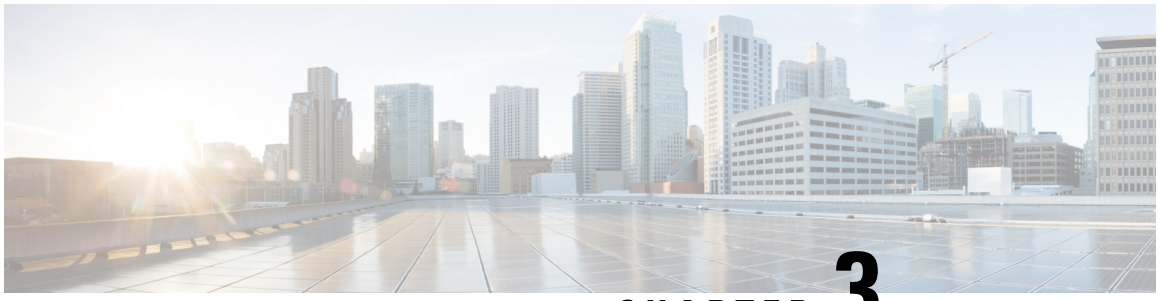
- Setup and Upgrade Workflow, on page 7

Setup and Upgrade Workflow

The workflow provides a high-level view of the steps involved in the setup and upgrade process. This workflow helps you in planning the tasks and minimizing the risk of errors or downtime.

Figure 2: Workflow to Setup and Upgrade the Router

With an understanding of this end-to-end workflow, you can get started with setting up and upgrading the IOS XR software on your Cisco 8000 series routers.



CHAPTER 3

Setup the Router

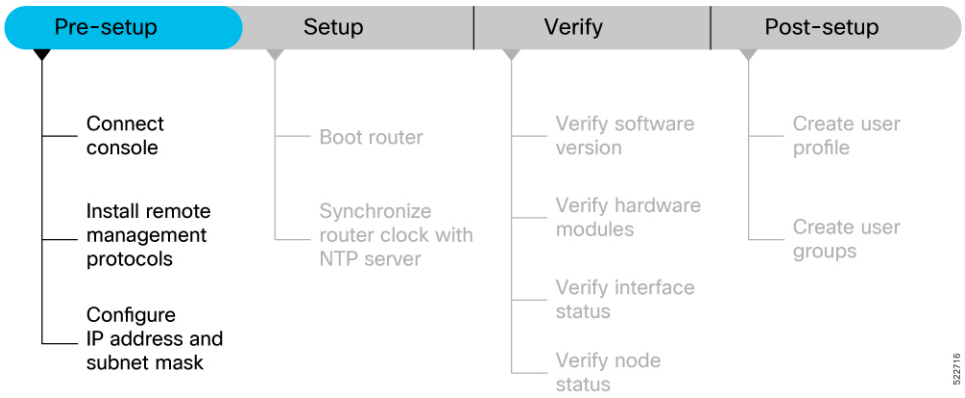
By following the guidelines provided on this page, you can set up the Cisco 8000 series routers quickly and efficiently.

- Prerequisites to Setup Router, on page 11
- Setup the Router, on page 13
- Verify the Software and Hardware Status, on page 17
- Complete Post-setup Tasks, on page 24

Prerequisites to Setup Router

Complete the following prerequisite tasks to prepare the router for seamless setup.

Figure 3: Pre-setup Workflow for the Cisco 8000 Series Routers



This section contains the following topics:

Connect Console Port to Terminal

The console port on the router is used to log into a router directly without a network connection using a terminal emulation program like HyperTerminal.

Procedure

Step 1 Connect the router to a terminal.

a) Locate the console port on the router.

Figure 4: Connect the Router to a Terminal

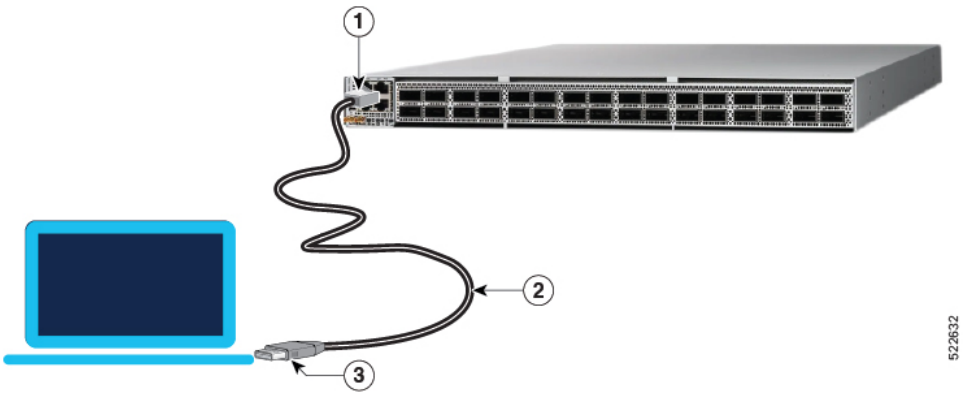


Table 3: Console Port and Cable Specifications

1	Routers console port
2	RJ-45 Rollover cable
3	• RJ-45/DSUB R/P adapter • RJ-45F/DB9F adapter • RJ-45/DSUB F/F adapter

b) Connect the console (or rollover) cable to the console port on the router.

c) Use the correct adapter to connect the other end of the cable to your terminal or PC.

Step 2 Configure the console port to match the following default port characteristics.

a) Launch the terminal session.

b) In the **COM1 Properties** window, select **Port Settings** tab, and enter the following settings:

- Speed – 115200
- Data Bits – 8
- Parity – none
- Stop bits – 1
- Flow Control – none

Step 3 Click **OK**.

You should see a blinking cursor in the HyperTerminal window indicating successful connection to the console port.

Install Remote Management Protocols

The router can be accessed using remote management protocols, such as SSH, SCP, FTP, and Telnet. The SSH, SCP, and FTP management protocols are included in the ISO image by default. Telnet is an optional package.

Procedure

Install the remote management protocols.

To install Telnet, you can use either of the following options:

- Install telnet package from the local directory of your router. The path to the local directory must be under `/harddisk:/` location. The following example shows how you can install the `xr-telnet-7.0.11v1.0.1-1.x86_64.rpm` optional package:

  ```
  Router#install source /harddisk:/files xr-telnet-7.0.11v1.0.1-1.x86_64.rpm
  ```

- Install telnet package from a configured repository.

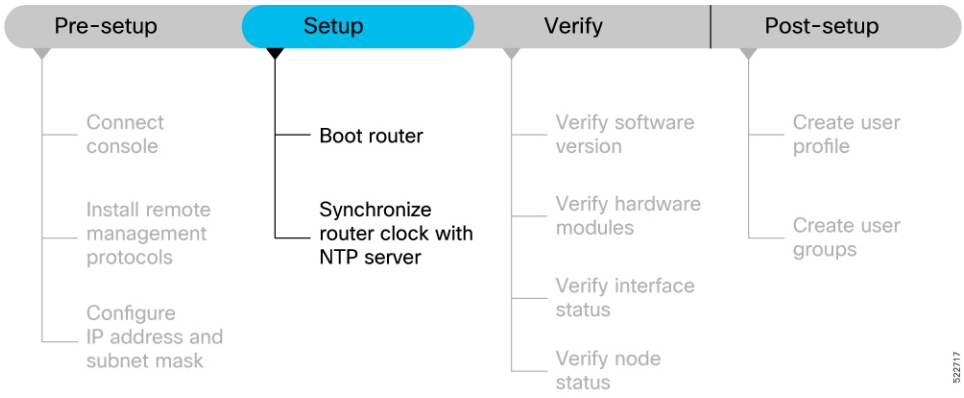
  ```
  Router#install source install-repo xr-telnet
  ```

  For information on creating and accessing an external or local repository, see Create Repository to Access Install Files, on page 43.

Setup the Router

Complete the following tasks to bring up your router for further configurations.

Figure 5: Setup Workflow for the Cisco 8000 Series Router



Boot the Router

After installing the hardware and connecting the console port to the terminal, boot the Cisco 8000 series router. The router completes the boot process using the pre-installed operating system image.

Before you begin

Ensure that you have completed the Prerequisites to Setup Router, on page 11.

Procedure

Step 1 Power ON the router.

The router completes the boot process using the pre-installed operating system image. If the router is not pre-installed with an image, you can boot the router using PXE boot an externally bootable USB drive or PXE boot.

Note
Effective Cisco IOS XR Software Release 7.10.1, we have updated the bootup logs on the router to display the information of Secure Shell Daemon (SSHD) process. The SSHD process is a part of the Cisco IOS XR ISO installation service.

The updated bootup log message is:

```
Starting IOS-XR ISO Installation including sshd...

[  OK ] IOS-XR ISO Installation including sshd.
```

Step 2 After booting is complete, follow the prompt to create a username and password. This credential is used to log on to the IOS XR console and get to the router prompt. The following prompt appears:

```
!!!!!!!!!!!!!!!!!!!! NO root-system username is configured. Need to configure root-system username.
 !!!!!!!!!!!!!!!!!!!!

--- Administrative User Dialog ---


Enter root-system username:
% Entry must not be null.

Enter root-system username: cisco
Enter secret:
Use the 'configure' command to modify this configuration.
User Access Verification

Username: cisco
Password:
```

See the Recover router from boot failure, on page 166 topic to resolve any boot failure issues.

Configure IP Address and Subnet Mask

Configure the IP address and subnet mask. The IP address and subnet mask for the Management Ethernet interface is used by the router for system management and remote communication.

Consult your network administrator or system planner to procure IP addresses and a subnet mask for the management interface.

Note We recommend that you use a Virtual Private Network (VPN) routing and VPN Routing and Forwarding (VRF) on the Management Ethernet interface.

Procedure

Step 1 Configure the IP address and a subnet mask for the Management Ethernet interface.

a) Configure the VRF.

Example:

```
Router(config)#vrf vrf1
Router(config-vrf)#exit
```

b) Configure the Management Ethernet Interface and set the VRF and IP address.

Example:

```
Router(config)#interface MgmtEth0/RSP0/CPU0/0
Router(config)#vrf vrf1
Router(config-if)#ipv4 address 10.10.0.1 255.0.0.0
Router(config-if)#ipv4 virtual address vrf vrf1 10.10.0.1/8
```

Configure multiple interfaces in a similar way.

c) Ensure that all available interfaces are discovered, and they in UP state.

Example:

```
Router(config-if)#no shutdown
Router(config-if)#exit
```

d) Configure a static route for communications with devices on other networks. Specify the IP address of the default gateway.

Example:

```
Router(config)#router static vrf vrf1 address-family ipv4 unicast 0.0.0.0/0 10.10.0.1
Router(config)#commit
```

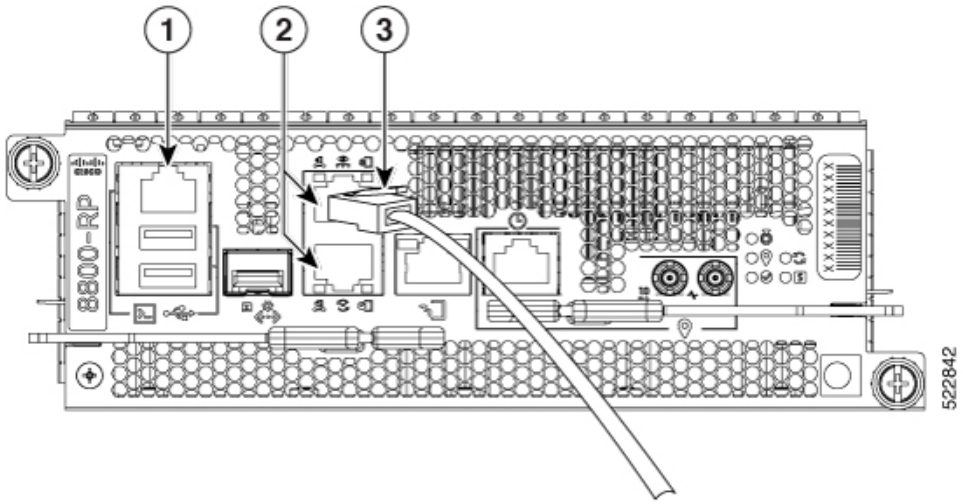
e) SSH into the management port.

Example:

```
Router#conf t
Router(config)#ssh server v2
Router(config)#commit
```

Step 2 Connect the management port to the Ethernet network. The physical port **Ethernet 0** on route processor is the management port.

Figure 6: Console Port and Management Ethernet Port



1	Console RS-232 Serial Port RJ-45
2	Management Ethernet Port (10/100/1000-Mbps) RJ-45 (Copper) port
3	Management Port connected to the Ethernet network

Example:

```
Server# ssh root@10.10.0.1
/etc/ssh/ssh_config line 18: Unsupported option "rhostsrsaauthentication"
/etc/ssh/ssh_config line 19: Unsupported option "rsaauthentication"
Warning: Permanently added 'x.xx.xx.xxx' (ECDSA) to the list of known hosts.
Password:
```

Synchronize Router Clock with NTP Server

You must synchronize the IOS XR clock with the Network Time Protocol (NTP) server to avoid a deviation from true time.

NTP uses the concept of a stratum to describe how many NTP hops away a machine is from an authoritative time source. A stratum 1 time server typically has an authoritative time source (such as a radio or atomic clock, or a GPS time source) directly attached to the server. A stratum 2-time server receives its time through NTP from a stratum 1 time server, and so on.

Note Cisco's implementation of NTP does not support stratum 1 service, and it is not possible to connect to a radio or atomic clock. We recommend that you obtain the time service for your network from the public NTP servers available on the IP Internet.

Procedure

Step 1 Synchronize the IOS XR clock with NTP server by going through the following example.

Example:

The NTP source is an IP address

```
Router(config)#ntp server NTP-source-IP-address
```

Example of NTP source is an IPv4 address:

```
Router(config)#ntp server 192.0.2.0
```

Example of NTP source is an IPv6 address:

```
Router(config)#ntp server 2001:DB8::1
```

Step 2 Commit the configuration.

Example:

```
Router(config-ntp)#commit
```

Step 3 Verify that the clock is synchronised with the NTP server.

Example:

```
Router#show ntp status
Clock is synchronized, stratum 3, reference is 192.0.2.0 nominal freq is 1000000000.0000 Hz,
actual freq is 1000000000.0000 Hz, precision is 2**24 reference time is E12B1B02.8BB13A2F
(08:42:42.545 UTC Tue Sep 17 2019) clock offset is -3.194 msec, root delay is 4.949 msec
root dispersion is 105.85 msec, peer dispersion is 2.84 msec loopfilter state is 'FREQ'
(Drift being measured), drift is 0.0000000000 s/s system poll interval is 64, last update
was 124 sec ago authenticate is disabled
```

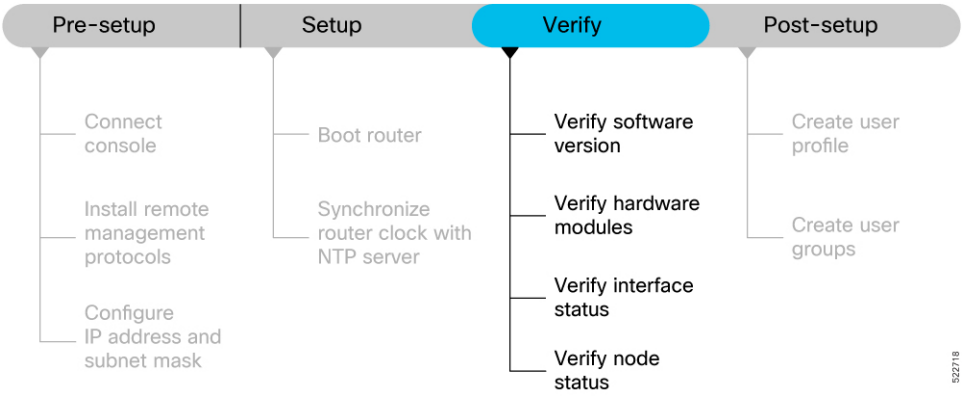
What to do next

Your router is now setup sucessfully. Perform prelimiary checks on the router to verify that the hardware and software components are functional.

Verify the Software and Hardware Status

After logging into the console, perform preliminary checks to verify the default setup.

Figure 7: Verification Workflow for the Cisco 8000 Series Router Setup



Ensure that you have completed the procedures in Setup the Router, on page 13 section before proceeding with the following verification tasks:

Verify Software Version

View the software version installed on the router.

Procedure

Verify the latest version of the Cisco IOS XR software installed on the router.

Example:

```
Router#show version
Build Information:
Built By : user1
Built On : Thu Feb 02 10:06:56 UTC 2023
Build Host : host
Workspace : /ws
Version : 7.8.1
Label : 7.8.1
```

Note
You must upgrade the system if a new version of the system is available to avail the latest features on the router.

For more information about upgrading the software version, see Upgrade the Router, on page 29.

Verify Hardware Modules

Cisco 8000 series routers have various hardware modules such as route processors, line cards, fan trays, and power modules installed on the router. Ensure that the firmware on various hardware components of the router is compatible with the installed Cisco IOS XR image. You also must verify that all the installed hardware and firmware modules are operational.

Procedure

Step 1 Verify the status of the hardware modules using the **show platform** command.

Example:

```
Router#show platform
Node Type State Config state
--------------------------------------------------------------
0/RP0/CPU0 8201-SYS(Active) IOS XR RUN NSHUT
Provision Network Devices using Zero Touch Provisioning
24
0/RP0/BMC0        8201-SYS            OPERATIONAL         NSHUT
0/PM0             PSU2KW-ACPE         OPERATIONAL         NSHUT
0/PM1             PSU2KW-ACPE         OPERATIONAL         NSHUT
0/FT0             FAN-1RU-PE          OPERATIONAL         NSHUT
0/FT1             FAN-1RU-PE          OPERATIONAL         NSHUT
0/FT2             FAN-1RU-PE          OPERATIONAL         NSHUT
0/FT3             FAN-1RU-PE          OPERATIONAL         NSHUT
0/FT4             FAN-1RU-PE          OPERATIONAL         NSHUT
```

Step 2 View the list of hardware and firmware modules that are detected on the router using the **show hw-module fpd** command.

Example:

```
Router#show hw-module fpd
FPD Versions
=================
Location         Card type     HWver       FPD device      ATR     Status       Running      Programd
-------------------------------------------------------------------------------------------------------
0/RP0/CPU0       8800-RP        0.51       Bios             S       CURRENT      1.15         1.15
0/RP0/CPU0       8800-RP        0.51       BiosGolden       BS      CURRENT      1.15
0/RP0/CPU0       8800-RP        0.51       BmcFitPrimary    S       NEED UPGD    0.240        0.240
0/RP0/CPU0       8800-RP        0.51       BmcFpga          S       NEED UPGD    0.18         0.18
0/RP0/CPU0       8800-RP        0.51       BmcFpgaGolden    BS      CURRENT      0.19
0/RP0/CPU0       8800-RP        0.51       BmcTamFw         S       CURRENT      5.05         5.05
0/RP0/CPU0       8800-RP        0.51       BmcTamFwGolden   BS      CURRENT      5.05
0/RP0/CPU0       8800-RP        0.51       BmcUbootPrimary S       CURRENT      0.15         0.15
0/RP0/CPU0       8800-RP        0.51       EthSwitch                CURRENT      0.07         0.07
0/RP0/CPU0       8800-RP        0.51       EthSwitchGolden BP      CURRENT      0.07
0/RP0/CPU0       8800-RP        0.51       TimingFpga               CURRENT      0.11         0.11
0/RP0/CPU0       8800-RP        0.51       TimingFpgaGolden B      CURRENT      0.11
0/RP0/CPU0       8800-RP        0.51       x86Fpga          S       NEED UPGD    0.23         0.23
0/RP0/CPU0       8800-RP        0.51       x86FpgaGolden    BS      CURRENT      0.24
0/RP0/CPU0       8800-RP        0.51       x86TamFw         S       CURRENT      5.05         5.05
0/RP0/CPU0       8800-RP        0.51       x86TamFwGolden   BS      CURRENT      5.05
```

From the **show hw-module fpd** output, verify that all hardware modules that are installed on the chassis are listed. An unlisted module indicates that the module is either malfunctioning, or has not been installed properly. You must remove and reinstall the hardware module.

The fields in the **show hw-module fpd** output are:

- **FPD Device**: Name of the hardware component, such as IO FPGA, IM FPGA, or BIOS. The Golden FPDs are not field upgradable.
- **Running**: Current version of the firmware running on the FPD.
- **Programd**: Version of the FPD programmed on the module
- **Status**: Upgrade status of the firmware. The different states are:

Table 4: Status and Description of the Firmware Upgrade

Status	Description
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 new firmware version is available in the installed image. We recommend that you to perform an upgrade of the firmware version.
RLOAD REQ	The upgrade is complete, 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 corrupt. Reinstall the firmware.
UPGD SKIP	The upgrade is skipped because the installed firmware version is higher than the one available in the image.

Step 3 Upgrade the required firmware as required, using the **upgrade hw-module location all fpd all** command.

Example:

```
Router#upgrade hw-module location all fpd all
Alarms are created showing all modules that needs to be upgraded.

Active Alarms
------------------------------------------------------------------------------------------------------------
Location     Severity  Group      Set Time                  Description
------------------------------------------------------------------------------------------------------------
0/6/CPU0     Major     FPD_Infra  09/16/2019 12:34:59 UTC   One Or More FPDs Need Upgrade Or Not In
 Current State
0/10/CPU0    Major     FPD_Infra  09/16/2019 12:34:59 UTC   One Or More FPDs Need Upgrade Or Not In
 Current State
0/RP0/CPU0   Major     FPD_Infra  09/16/2019 12:34:59 UTC   One Or More FPDs Need Upgrade Or Not In
 Current State
0/RP1/CPU0   Major     FPD_Infra  09/16/2019 12:34:59 UTC   One Or More FPDs Need Upgrade Or Not In
 Current State
0/FC0        Major     FPD_Infra  09/16/2019 12:34:59 UTC   One Or More FPDs Need Upgrade Or Not In
 Current State
0/FC1        Major     FPD_Infra  09/16/2019 12:34:59 UTC   One Or More FPDs Need Upgrade Or Not In
 Current State
```

Note
The BIOS and IOFPGA upgrades require a restart of the router for the new version to take effect.

Step 4 Verify status of the modules after upgrade using the **show hw-module fpd** command.

Example:

```
Router#show hw-module fpd
Location      Card type     HWver      FPD device     ATR        Status       Running     Programd
------------------------------------------------------------------------------------------------
0/RP0/CPU0    8800-RP       0.51       Bios             S          CURRENT    1.15        1.15
0/RP0/CPU0    8800-RP       0.51       BiosGolden       BS         CURRENT    1.15
0/RP0/CPU0    8800-RP       0.51       BmcFitPrimary    S          RLOAD REQ  0.240       0.241
0/RP0/CPU0    8800-RP       0.51       BmcFpga          S          RLOAD REQ  0.18        0.19
0/RP0/CPU0    8800-RP       0.51       BmcFpgaGolden    BS         CURRENT    0.19
0/RP0/CPU0    8800-RP       0.51       BmcTamFw         S          CURRENT    5.05        5.05
```

```
0/RP0/CPU0    8800-RP       0.51       BmcTamFwGolden    BS         CURRENT    5.05
0/RP0/CPU0    8800-RP       0.51       BmcUbootPrimary   S          CURRENT    0.15       0.15
0/RP0/CPU0    8800-RP       0.51       EthSwitch                    CURRENT    0.07       0.07
0/RP0/CPU0    8800-RP       0.51       EthSwitchGolden   BP         CURRENT    0.07
0/RP0/CPU0    8800-RP       0.51       TimingFpga                   CURRENT    0.11       0.11
0/RP0/CPU0    8800-RP       0.51       TimingFpgaGolden  B          CURRENT    0.11
0/RP0/CPU0    8800-RP       0.51       x86Fpga           S          RLOAD REQ  0.23       0.24
0/RP0/CPU0    8800-RP       0.51       x86FpgaGolden     BS         CURRENT    0.24
0/RP0/CPU0    8800-RP       0.51       x86TamFw          S          CURRENT    5.05       5.05
0/RP0/CPU0    8800-RP       0.51       x86TamFwGolden    BS         CURRENT    5.05
```

The status of the upgraded nodes shows that a reload is required.

Step 5 Reload the individual nodes that require an upgrade.

Example:

```
Router#reload location node-location
```

Step 6 Verify that all nodes that had required an upgrade now shows an updated status of CURRENT with an updated FPD version.

Example:

```
Router#show hw-module fpd
FPD Versions
=================
Location     Card type   HWver   FPD device      ATR      Status       Running     Programd
-------------------------------------------------------------------------------------------
0/RP0/CPU0   8800-RP     0.51     Bios            S        CURRENT      1.15        1.15
0/RP0/CPU0   8800-RP     0.51     BiosGolden      BS       CURRENT      1.15
0/RP0/CPU0   8800-RP     0.51     BmcFitPrimary   S        CURRENT      0.241       0.241
0/RP0/CPU0   8800-RP     0.51     BmcFpga         S        CURRENT      0.19        0.19
0/RP0/CPU0   8800-RP     0.51     BmcFpgaGolden   BS       CURRENT      0.19
0/RP0/CPU0   8800-RP     0.51     BmcTamFw        S        CURRENT      5.05        5.05
0/RP0/CPU0   8800-RP     0.51     BmcTamFwGolden  BS       CURRENT      5.05
0/RP0/CPU0   8800-RP     0.51     BmcUbootPrimary S        CURRENT      0.15        0.15
0/RP0/CPU0   8800-RP     0.51     EthSwitch                CURRENT      0.07        0.07
0/RP0/CPU0   8800-RP     0.51     EthSwitchGolden BP       CURRENT      0.07
0/RP0/CPU0   8800-RP     0.51     TimingFpga               CURRENT      0.11        0.11
0/RP0/CPU0   8800-RP     0.51     TimingFpgaGolden B       CURRENT      0.11
0/RP0/CPU0   8800-RP     0.51     x86Fpga          S       CURRENT      0.24        0.24
0/RP0/CPU0   8800-RP     0.51     x86FpgaGolden    BS      CURRENT      0.24
0/RP0/CPU0   8800-RP     0.51     x86TamFw         S       CURRENT      5.05        5.05
0/RP0/CPU0   8800-RP     0.51     x86TamFwGolden   BS      CURRENT      5.05
```

Note
For more information on upgrading FPDs, see the Upgrading Field-Programable Device chapter.

Verify Interface Status

All available interfaces must be discovered by the system after booting the Cisco 8000 Series Router. Interfaces not discovered might indicate a malfunction in the unit.

Procedure

Use the **show ipv4 interfaces brief** or **show ipv6 interfaces brief** command to view the interfaces discovered by the system.

Example:

```
Router#show ipv4 interfaces brief
Interface             IP-Address        Status         Protocol      Vrf-Name
-------------------------------------------------------------------------------
HundredGigE0/0/0/0    unassigned        Shutdown       Down          default
HundredGigE0/0/0/1    unassigned        Shutdown       Down          default
HundredGigE0/0/0/2    unassigned        Shutdown       Down          default
HundredGigE0/0/0/3    unassigned        Shutdown       Down          default
HundredGigE0/0/0/4    unassigned        Shutdown       Down          default
HundredGigE0/0/0/5    unassigned        Shutdown       Down          default
HundredGigE0/0/0/6    unassigned        Shutdown       Down          default
HundredGigE0/0/0/7    unassigned        Shutdown       Down          default
------------------------------- <snip> -------------------------------------
TenGigE0/0/0/18/0     unassigned        Up             Up            default
TenGigE0/0/0/18/1     unassigned        Up             Up            default
TenGigE0/0/0/18/2     unassigned        Up             Up            default
TenGigE0/0/0/18/3     unassigned        Up             Up            default
MgmtEth0/RP0/CPU0/0   10.10.10.1        Up             Up            default
```

When a router is turned ON for the first time, all interfaces are in the **unassigned** state.

Ensure that the total number of interfaces that are displayed in the result matches with the actual number of interfaces present on the router, and that the interfaces are created according to the type of line cards displayed in **show platform** command.

Verify Node Status

A node can be a specified location, or the complete hardware module in the system. You must verify that the software state of all route processors, line cards, and the hardware state of fabric cards, fan trays, and power modules are listed, and their state is OPERATIONAL. This indicates that the IOS XR console is operational on the cards.

Procedure

Verify the operational status of the node using the **show platform** command.

Example:

```
Router#show platform
Node          Type                 State          Config state
--------------------------------------------------------------
0/RP0/CPU0    8800-RP(Active)      IOS XR RUN     NSHUT
0/RP0/BMC0    8800-RP              OPERATIONAL    NSHUT
0/RP1/CPU0    8800-RP(Standby)     IOS XR RUN     NSHUT
0/RP1/BMC0    8800-RP              OPERATIONAL    NSHUT
0/0/CPU0      8800-LC              IOS XR RUN     NSHUT
0/11/CPU0     8800-LC              IOS XR RUN     NSHUT
```

```
0/FC0          8800-FC              OPERATIONAL    NSHUT
0/FC3          8800-FC              OPERATIONAL    NSHUT
0/FT0          8800-FAN             OPERATIONAL    NSHUT
0/FT1          8800-FAN             OPERATIONAL    NSHUT
0/FT2          8800-FAN             OPERATIONAL    NSHUT
0/FT3          8800-FAN             OPERATIONAL    NSHUT
0/PT0          FAM7000-ACHV-TRAY    OPERATIONAL    NSHUT
```

Table 5: Card Type, Node Status, and Description

Card Type	State	Description
All	UNKNOWN	Error – Internal card record is not available
All	IDLE	Error – Card state is not initialized
All	DISCOVERED	Card is detected
All	POWERED_ON	Card is powered on
RP, LC	BIOS_READY	Card BIOS is up
RP, LC	IMAGE_INSTALLING	Image is being downloaded or installed
RP, LC	BOOTING	Image is installed and the software is booting up
RP, LC	IOS_XR_RUN	Software is operating normally and is functional
RP, LC	IOS_XR_INITIALIZING	Software is initializing
FC, FT, PT, PM	OPERATIONAL	Card is operating normally and is functional
RP, LC, FC	RESET	Card is undergoing reset
RP, LC	REIMAGE	Card is pending reimage
RP, LC, FC	SHUTTING_DOWN	Card is shutting down as a result of a fault condition, user action or configuration
RP, LC, FC	SHUT_DOWN	Card is shutdown due to a fault condition, user action or configuration
FC	ONLINE	RP is able to access this remote card
LC	DATA_PATH_POWERED_ON	Forwarding complex is powered ON
RP (Active)	SHUTTING_REMOTE_CARDS	Active RP card is in the process of shutting down other cards as part of a chassis reset
RP (Standby), LC, FC	WAITING_FOR_CHASSIS_RESET	Card is shutdown and is waiting for the chassis to be reset

Card Type	State	Description
RP, LC	WDOG_STAGE1_TIMEOUT	Card CPU failed to reset the hardware watchdog
RP, LC	WDOG_STAGE2_TIMEOUT	Hardware watchdog has timed out waiting for the card CPU to reset itself
RP, LC, FC	FPD_UPGRADE	One or more FPD upgrades are in progress
FC	CARD_ACCESS_DOWN	RP is unable to access this remote card
RP (standby only), LC	BOOT HOLD	In a multinode system, any node reloads that occur during a transaction that are not initiated as part of the installation shows a BOOT HOLD state. The node continues to be in this state until the transaction is either committed or cancelled

What to do next

This completes verification of the basic router setup. You can now complete the post-setup tasks where you manage user profiles and groups.

Complete Post-setup Tasks

You must create user profiles and user groups to manage your system, install software packages, and configure your network.



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.

Every user is authenticated using a username and a password. The authentication, authorization, and accounting (AAA) commands help with these services:

- Create users, groups, command rules, or data rules
- Change the disaster-recovery password

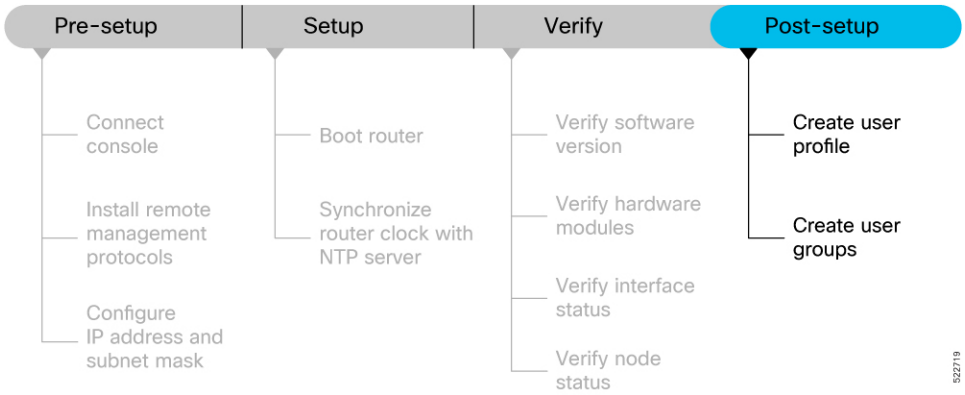
IOS-XR and Linux have separate AAA services and IOS XR AAA is the primary AAA system. A user who is created through IOS-XR can log in directly to the EXEC prompt when connected to the router, while a user created through Linux can connect to the router, but can log in to the bash prompt. The user must log in to IOS XR explicitly, to access the IOS-XR EXEC prompt.

You must configure the IOS-XR AAA authorization to restrict users from uncontrolled access. If AAA is not configured, the command and data rules associated to the groups that are assigned to the user are ignored. A

user can have full read/write access to IOS XR configuration through Network Configuration Protocol (NETCONF), google-defined Remote Procedure Calls (gRPC), or any YANG-based agents. To avoid granting uncontrolled access, enable AAA before setting up any configuration. To gain an understanding about AAA, and to explore the AAA services, see the *Configuring AAA Services* chapter in the *System Security Configuration Guide for Cisco 8000 Series Routers*.

The following image provides you an overview of the various tasks that are involved in the Cisco 8000 Series Routers post-setup procedure.

Figure 8: Post-setup Workflow for the Cisco 8000 Series Router



Ensure that you have completed the Setup the Router, on page 13 and Verify the Software and Hardware Status, on page 17 tasks before you perform the following tasks:

Create User Profile

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.

Perform the following steps to create a user profile:

Procedure

Step 1 Create a user, provide a password and assign the user to a group. For example, **user1** is the user, password is **pw123**, and the group is **root-lr**.

Example:

```
Router#config

/* Create a new user */
Router(config)#username user1

/* Set a password for the new user */
Router(config-un)#password pw123

/* Assign the user to group root-lr */
Router(config-un)#group root-lr
```

All users have read privileges. The **root-lr** users inherit write privileges where users can create configurations, create new users, and so on.

Enable display of login banner: The US Department of Defense (DOD)-approved login banner provides information such as number of successful and unsuccessful login attempts, time stamp, login method, and so on. 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.

You can enable or diable the login login banner by using the **login-history enable** and **login-history disable** commands.

Note
Login notifications get reset during a router reload.

Step 2 Run the **show running-config username user1** command to verify the state of login banner.

Example:

```
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 3 Commit the configuration.

Example:

```
Router(config-un)#commit
```

The user profile is created and allowed access to the router based on the configured privileges.

Create User Groups

You can 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. The router supports a maximum of 32 user groups.

Before you begin

Ensure that you have created a user profile. See Create User Profile, on page 25.

Procedure

Step 1 Create a new user group.

Example:

```
Router#config

/* Create a new user group, group1 */
Router#(config)#group group1

/* Specify the name of the user, user1 to assign to this user group */
Router#(config-GRP)#username user1
```

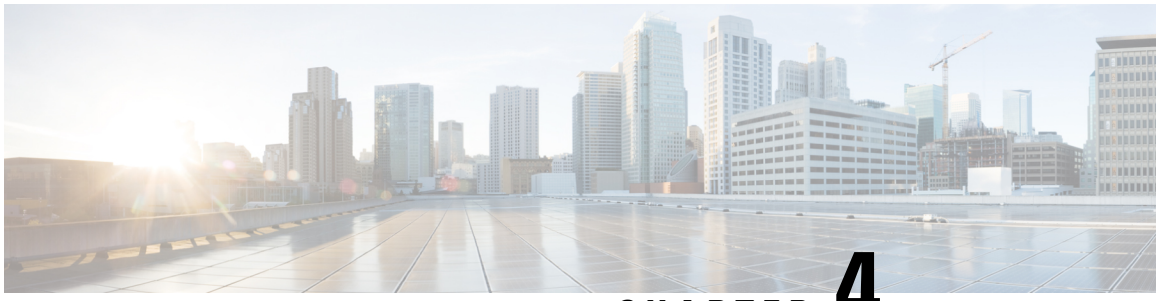
Step 2 Commit the configuration.

Example:

```
Router(config-GRP)#commit
```

What to do next

This completes the router setup and verification process. You can now proceed with upgrading the software, installing RPMs, SMUs and bug fixes based on your requirement.



CHAPTER 4

Upgrade the Router

Your Cisco router comes preinstalled with IOS XR software. You can upgrade the router by installing a new version of the software. We recommend that you keep the software up-to-date to ensure that the router works with the latest features and bug fixes.

During an upgrade:

- the newer software replaces the currently active software on the router.
- packages (RPMs) that have the same name and version in the current and target release versions are not removed or reinstalled.

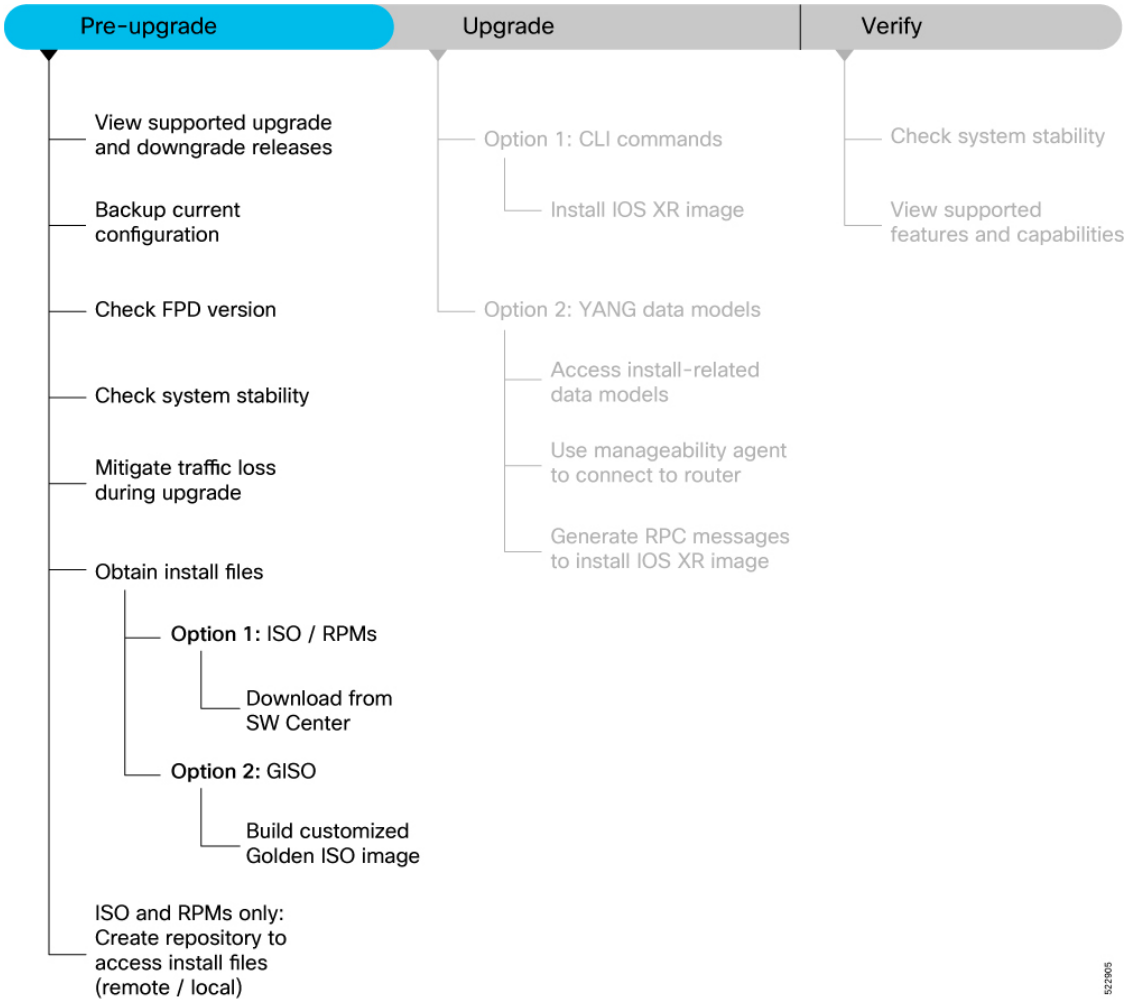
The following image shows the tasks involved in successfully upgrading the router.

- Plan the Software Upgrade , on page 29
- Upgrade the Software, on page 51
- Verify the Software Upgrade, on page 61
- Optics firmware upgrades, on page 67
- Deprecated commands, on page 70

Plan the Software Upgrade

Before you upgrade the software version, prepare the router to ensure that the upgrade process is seamless.

Figure 9: Pre-upgrade Workflow for the Cisco 8000 Series Router



This section describes the following processes to prepare your router for an upgrade:

Supported upgrade and downgrade releases

Cisco routers come preinstalled with Cisco IOS XR software. You can upgrade the software to access new features and fixes, or downgrade it if needed. To take advantage of the latest features and software improvements, we recommend that you keep your router updated with the current version.

Table 6: Feature History Table

Feature Name	Release Information	Description
Optimized IOS XR upgrade or downgrade process	Release 25.3.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q200, P100], 8700 [ASIC: P100, K100], 8010 [ASIC: A100]); Centralized Systems (8600 [ASIC:Q200]) ; Modular Systems (8800 [LC ASIC: Q100, Q200, P100]) You can now upgrade Cisco IOS XR software from an earlier version to version 25.3.1 or later, and downgrade from version 25.3.1 or later to an earlier version, with certain limitations. These limitations are designed to help prevent failures during the upgrade or downgrade process.
Changes to supported software upgrade or downgrade IOS-XR versions due to disk encryption	Release 25.2.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q200, P100], 8700 [ASIC: P100, K100], 8010 [ASIC: A100]); Centralized Systems (8600 [ASIC:Q200]) ; Modular Systems (8800 [LC ASIC: Q100, Q200, P100]) Cisco IOS XR software versions that are supported on the router can now be upgraded to or dowgraded from version 25.2.1 or later within the scope and limitations of the disk encryption feature introduced in this release.
View 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.

For release specific caveats see Release-specific Caveats and Workarounds, on page 207 section.

Prerequisites

Before you begin the upgrade, you must identify a Cisco IOS XR release that aligns with Cisco-recommended upgrade paths.

Use the **show install upgrade-matrix running** command to identify a supported target upgrade release, and prerequisites or limitations related to the specific software upgrade or downgrade. This command provides the following information:

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

Supported software upgrade from running version

Example: In this example, the output of the **show install upgrade-matrix running** command displays the upgrade restrictions.

```
Router#show install upgrade-matrix running
Matrix: XR version: 7.9.1, File version: 1.0
The upgrade matrix indicates that the following system upgrades are supported from the
current XR version:

From                To              Restrictions
7.9.1               7.7.1           CSCab54345
7.9.1               7.7.2           -
7.9.1               7.7.3           -
7.9.1               7.7.4           -
7.9.1               7.7.5           -
7.9.1               7.7.6           -
7.9.1               7.8.1           -
```

Example: This example shows all supported releases for upgrade from the current version on the Cisco 8000 routers:

```
Router# show install upgrade-matrix running
Wed Aug  6 13:40:20.655 UTC
Matrix: XR version: 25.3.1, File version: 1.0, Version: N/A


The upgrade matrix indicates that the following system upgrades are supported from the
current XR version:


From       To         Restrictions
---------- ---------- ------------------------------------------------------
25.3.1     24.1.2     Target fixes; Caveats; Replace performed via reimage
25.3.1     24.2.11    Target fixes; Caveats; Replace performed via reimage
25.3.1     24.2.2     Target fixes; Caveats; Replace performed via reimage
25.3.1     24.2.20    Caveats; Replace performed via reimage
25.3.1     24.2.200   Caveats; Replace performed via reimage
25.3.1     24.2.21    Caveats; Replace performed via reimage
25.3.1     24.3.1     Target fixes; Caveats; Replace performed via reimage
25.3.1     24.3.2     Target fixes; Caveats; Replace performed via reimage
25.3.1     24.3.20    Caveats; Replace performed via reimage
25.3.1     24.3.30    Caveats; Replace performed via reimage
```

```
25.3.1     24.4.1     Caveats; Replace performed via reimage
25.3.1     24.4.2     Caveats; Replace performed via reimage
25.3.1     25.1.1     Caveats; Replace performed via reimage
25.3.1     25.1.2     Caveats; Replace performed via reimage
25.3.1     25.2.1     Caveats; Replace performed via reimage
25.3.1     25.2.15    Caveats; Replace performed via reimage
25.3.1     25.2.2     Caveats; Replace performed via reimage
25.3.1     7.10.2     Target fixes; Caveats; Replace performed via reimage
25.3.1     7.11.2     Target fixes; Caveats; Replace performed via reimage
25.3.1     7.11.21    Target fixes; Caveats; Replace performed via reimage


Add the from and to versions to the end of the CLI command, for data on versions with
additional restrictions


For example, to display restrictions for the 25.3.1->24.1.2 upgrade, use
        'show install upgrade-matrix running 25.3.1 24.1.2'



Router#
```

Supported releases to upgrade software from current version to target version

In this example, you provide the current version and the target version that you want to upgrade the router. The output of the command displays the support information and dependencies between these two releases:

```
Router#show install upgrade-matrix running 7.5.2 7.3.1
Tue May 10 19:33:59.135 UTC

Upgrade matrix information for system upgrade: 7.5.2->7.3.1

XR system upgrade is supported, with the following restrictions:

The following fixes must be installed if any version of the package is installed.
-----------------------------------------
Ddts              Name        Version
-----------------------------------------
CSCab54345        xr-bgp      7.5.2
```

Show commands for software upgrade and downgrade

You can view the software upgrade and downgrade information using the **show** commands in this table 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 running v1 v2`	Displays details about the software upgrades from version 1 to version 2 according to the support data installed on the running system
`show install upgrade-matrix running all`	Displays all supported software upgrades from any version according to the support data installed on the running system

Command	Description
`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 v1 v2`	Displays details about the software upgrade from version 1 to version 2 according to the support data in the target 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 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 From Cisco IOS XR Release 25.2.1, the **running** keyword is deprecated from the **show install upgrade-matrix iso** command and replaced with the **from-running** keyword. This is applicable for routers with Cisco IOS XR7 network operating systems.
`show install upgrade-matrix rollback`	Displays details about the software upgrade from the current version to a version of a specific rollback point (indicated by an ID) according to the support matrices in both the running system and the rollback ID
`show install upgrade-matrix rollback ID v1 v2`	Displays details about the software upgrade from version 1 to version 2 according to the support data in the specific rollback ID
`show install upgrade-matrix rollback ID all`	Displays all supported software upgrades from any version according to the support data in the specific rollback ID
`show install upgrade-matrix rollback running`	Displays details about the software upgrade from the current version to the version of the specific rollback ID according to the support matrices in both the running system and the rollback ID

Backup Current Configuration

The ability to recover from a disaster is an essential part of any system maintenance plan. We recommend you backup the configurations in a secure remote location and verify that the transfer is a success, both before and after upgrade.

Procedure

Step 1 Create a backup of the running configuration to one of the following locations based on your requirement:

- Copy the configuration to the `harddisk:` location on the router.

```
Router#copy running-config harddisk:/running_config-<mmddyyyy>
Destination filename [running_config-<mmddyyyy>]?
Building configuration...
[OK]
Verifying checksum...  OK (0xDCF1)
```

- Copy the configuration to a remote server. Ensure the router has root access to the server.

```
Router#scp harddisk:/ running_config-<mmddyyyy>
user:password@<ip-address>:<location>
```

Step 2 Verify that the configuration is backed up.

Check FPD Version

The router uses a number of Field Programmable Devices (FPDs) that are crucial for the function of route processors, line cards, shared port adapters (SPAs), SPA Interface Processors (SIPs), and fan trays. Before upgrading the software, check whether the latest FPDs are available on the router.

Note FPD auto-upgrade is enabled by default on the Cisco 8000 series routers. However, we recommend that when updating to IOS XR Release 7.5.1, configure the **fpd auto-upgrade enable** command.

```
Router#show hw-module fpd

                                                                          FPD Versions
                                                                          ================
Location       Card type   HWver    FPD device        ATR  Status      Running   Programd
-------------------------------------------------------------------------------------------
0/RP0/CPU0     8800-RP     0.51     Bios              S    CURRENT     1.15      1.15
0/RP0/CPU0     8800-RP     0.51     BiosGolden        BS   CURRENT     1.15

0/RP0/CPU0     8800-RP     0.51     EthSwitch              CURRENT     0.07      0.07
0/RP0/CPU0     8800-RP     0.51     EthSwitchGolden   BP   CURRENT     0.07
0/RP0/CPU0     8800-RP     0.51     TimingFpga             CURRENT     0.11      0.11
0/RP0/CPU0     8800-RP     0.51     TimingFpgaGolden  B    CURRENT     0.11
0/RP0/CPU0     8800-RP     0.51     x86Fpga           S    NEED UPGD   0.23      0.23
0/RP0/CPU0     8800-RP     0.51     x86FpgaGolden     BS   CURRENT     0.24
0/RP0/CPU0     8800-RP     0.51     x86TamFw          S    CURRENT     5.05      5.05
0/RP0/CPU0     8800-RP     0.51     x86TamFwGolden    BS   CURRENT     5.05
```

In this example, `x86Fpga` FPD device needs an upgrade. You must ensure that FPDs are upgraded *before* upgrading the router.

Procedure

Step 1 To manually upgrade FPDs, use the **upgrade hw-module fpd** command.

```
Router#upgrade hw-module location all fpd all
Alarms are created showing all modules that needs to be upgraded.

Active Alarms
------------------------------------------------------------------------------------------------------------------
Location      Severity     Group        Set Time                    Description
------------------------------------------------------------------------------------------------------------------

0/6/CPU0       Major       FPD_Infra    09/16/2019 12:34:59 UTC     One Or More FPDs Need Upgrade Or
 Not In Current State
0/10/CPU0      Major       FPD_Infra    09/16/2019 12:34:59 UTC     One Or More FPDs Need Upgrade Or
 Not In Current State
0/RP0/CPU0     Major       FPD_Infra    09/16/2019 12:34:59 UTC     One Or More FPDs Need Upgrade Or
```

```
 Not In Current State
0/RP1/CPU0     Major       FPD_Infra    09/16/2019 12:34:59 UTC     One Or More FPDs Need Upgrade Or
 Not In Current State
0/FC0          Major       FPD_Infra    09/16/2019 12:34:59 UTC     One Or More FPDs Need Upgrade Or
 Not In Current State
0/FC1          Major       FPD_Infra    09/16/2019 12:34:59 UTC     One Or More FPDs Need Upgrade Or
 Not In Current State
```

Note
BIOS and IOFPGA upgrades require a power cycle of the router for the new version to take effect.

For example:

```
Router#upgrade hw-module location all fpd all
upgrade command issued (use "show hw-module fpd" to check upgrade status)
Router#
```

```
Router#show hw-module fpd
                                                                                FPD Versions
                                                                                ================
Location       Card type   HWver    FPD device        ATR  Status     Running   Programd
-------------------------------------------------------------------------------------------
0/RP0/CPU0     8800-RP     0.51     Bios              S    CURRENT    1.15      1.15
0/RP0/CPU0     8800-RP     0.51     BiosGolden        BS   CURRENT    1.15
0/RP0/CPU0     8800-RP     0.51     EthSwitch              CURRENT    0.07      0.07
0/RP0/CPU0     8800-RP     0.51     EthSwitchGolden   BP   CURRENT    0.07
0/RP0/CPU0     8800-RP     0.51     TimingFpga             CURRENT    0.11      0.11
0/RP0/CPU0     8800-RP     0.51     TimingFpgaGolden  B    CURRENT    0.11
0/RP0/CPU0     8800-RP     0.51     x86Fpga           S    RLOAD REQ  0.23      0.24
0/RP0/CPU0     8800-RP     0.51     x86FpgaGolden     BS   CURRENT    0.24
0/RP0/CPU0     8800-RP     0.51     x86TamFw          S    CURRENT    5.05      5.05
0/RP0/CPU0     8800-RP     0.51     x86TamFwGolden    BS   CURRENT    5.05
```

Step 2 Reload the individual nodes that require an upgrade by using the **reload location** *node-location* command.

For example:

```
Router#reload location 0/RP0
Proceed with reload?  [confirm]
```

Note
The system requests recovery reload by default when the system detects fault. However, if you want to prevent the recovery reload for debugging, use the **hw-module reset auto disable location** command to disable an auto reset mechanism. You can use the **hw-module reset auto disable location** command in global configuration mode.

If you want to re-enable the recovery reload, use the **no hw-module reset auto disable location** command.

Step 3 You can enable **automatic upgrade of FPD** by using the **fpd auto-upgrade enable** command.

To automatically upgrade all FPDs, use:

```
Router(config)#fpd auto-upgrade enable
```

Usage Guidelines—Online Insertion of IMs

When an IM **with a lower FPD version** is inserted, one of the following scenarios apply:

- If `fpd auto-upgrade` is enabled and a new IM is inserted, the system upgrades the IMs FPDs automatically with the latest FPDs.
- If `fpd auto-upgrade` is disabled, no action is required.

  Note

Cisco **recommends** enabling the `fpd auto-upgrade`. If you disable it, you must manually check the FPD upgrade on the individual nodes using the **show hw-module fpd** command and reload the individual nodes that require an upgrade using the **reload location** *node-location* command.

Usage Guidelines—Online Insertion of RPs

When **fpd auto-upgrade** is enabled and a new RP is inserted, the system upgrades the RP FPDs automatically with the latest FPDs.

Verify that all nodes that required an upgrade show an updated status of `CURRENT` with an updated FPD version using the **show hw-module fpd** command.

Note
For more information on upgrading FPDs, see the *Upgrading Field Programmable Device* chapter.

Upgrading FPDs Using Yang Data Models

YANG is a data modeling language that helps to create configurations, retrieve operational data and execute actions. The router acts on the data definition when these operations are requested using NETCONF RPCs. The data model handles the following types of requirements on the routers for FPD:

Operational Data	Native Data Model
Auto Upgrade: Enabling or disabling of automatic upgrade of FPD.	Cisco-IOS-XR-fpd-infra-cfg.yang

Check System Stability

System stability checks are essential to measure the efficiency and ability of an upgrade to function over an extended period.

At the EXEC prompt, execute the following commands to assess basic system stability checks before and after the software upgrade.

Command	Reason	Workaround
show platform	Verify that all nodes are in `IOS XR RUN/OPERATIONAL` state	NA
show redundancy	Verify that a standby RP is available, and the system is in `NSR-ready` state	NA
show ipv4 interface brief Or **show ipv6 interface brief** Or **show interfaces summary**	Verify that all necessary interfaces are `UP`	NA

Command	Reason	Workaround
show install active summary	Verify that the proper set of packages are active	NA
show install committed summary	Verify that the proper set of committed packages are same as active	Execute 'install commit' command
clear configuration inconsistency	Verify/fix configuration file system	NA
show hw-module fpd	Ensure all the FPD versions status are CURRENT	Execute `upgrade hw-module fpd` command
show media	Display the current state of the disk storage media	To free up space, remove older .iso image files and bug fix .tar files.
show media | i rootfs	Display the current state of the root filesystem (rootfs). By default, the following files are stored in **rootfs**: • Older config commits • Older .iso image and .tar files for SMUs • All the extracted .tar files	The installation is blocked if it utilizes more than 92% of the disk space on the **rootfs**. To avoid this, we recommend maintaining: • Twice the free space of the .iso image file size when installing the software • At least two and a half times the size of the .tar file when installing SMUs To free up space in **rootfs**: • use the **clear install rollback id** *id* to remove older rollback points • consider storing all user data in the **harddisk:/** location
show inventory	Show chassis inventory information	NA
show logging	Capture show logging to check for any errors	NA

Mitigate Traffic Loss During Upgrade

During an upgrade, any traffic routed through the device is affected. To minimize traffic loss during the upgrade, do the following:

For OSPF, configure the router to advertise a maximum metric so that other devices do not prefer the router as an intermediate hop in their SPF calculations:

```
Router(config-ospf)#max-metric router-lsa
```

For ISIS, set the overload bit for a fixed amount of time. This ensures that the router does not receive transit traffic while the routing protocol is still converging:

```
Router(config-isis)#set-overload-bit on-startup <timeout>
```

Optional packages

Cisco IOS XR software images are composed of a base image (ISO) that includes mandatory packages and optional packages (RPMs). Optional RPMs are software packages included in or available for the IOS XR image that provide additional features beyond the mandatory base functionality.

These optional RPMs can be included in the ISO image or provided separately for installation as needed. Optional RPMs enable specific features such as routing protocols such as OSPF, RIP, ISIS, and so on, security features such as k9sec for 802.1X, multicast, telemetry, and others.

Some optional RPMs are included in the base ISO image, while others are available as separate downloadable packages. Operators can add, upgrade, or remove optional RPMs using IOS XR software management commands, allowing flexible customization of the software features on the device.

Table 7: Feature History Table

Feature Name	Release Information	Description
Deprecated insecure protocols in optional RPMs	Release 26.2.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q200, P100], 8700 [ASIC: P100, K100], 8010 [ASIC: A100]); Centralized Systems (8600 [ASIC:Q200]) ; Modular Systems (8800 [LC ASIC: Q100, Q200, P100]) This feature improves platform security and reduces the threat surface of the Cisco Networking products by restricting the insecure protocol such as Telnet (deprecated in Cisco IOS XR Release 25.4.1) from Cisco IOS XR base package. In this release, Telnet is added to the optional RPM, `Telnet.rpm`. We recommend you to migrate to secure alternatives such as SSH as Telnet is not supported anymore in the Cisco IOS XR base package.

Starting with Cisco IOS XR Release 26.2.1, the insecure protocol such as Telnet is no longer part of Cisco IOS XR base packages. Instead, install the new optional RPM, `Telnet.rpm`, to use Telnet features.

You must install the `Telnet.rpm` only if it is obsolutely required as deprecation warnings continue to be generated once every 30 days as long as Telnet configurations are used. The support for `Telnet.rpm` will be completely removed from Cisco IOS XR in future releases such that Telnet will no longer be supported in Cisco IOS XR.

To avoid system failures or configuration inconsistencies, we strongly recommend you to migrate to a secure protocol such as SSHv2.

Limitations of optional packages

Without the optional `Telnet.rpm`:

- Telnet configuration and execution commands cannot be used.
- Telnet to Management ports is not supported.
- Systems upgraded with existing insecure configurations may raise configuration inconsistency alarms.

  You must clear inconsistencies before committing any new configuration.

Commands and YANG data models supported using optional RPMs

In Cisco IOS XR Release 26.2.1, you can continue to use the commands and YANG data models for the deprecated Telnet protocol if you have installed the new `Telnet.rpm`.

The following table provides the supported configuration and execution commands and the YANG data models for Telnet in `Telnet.rpm`.

Table 8: Supported Telnet configuration and execution commands in Telnet.rpm

Telnet config commands	Telnet exec commands
telnet ipv4 client source-interface <>	**telnet** <*ipv4-address*>
telnet ipv4 dscp <>	**telnet** <*ipv6-address*>
telnet ipv4 server max-servers <>	**telnet** <*hostname*>
telnet ipv6 client source-interface <>	**telnet disconnect-char** <*WORD*>
telnet ipv6 server max-servers <>	**telnet vrf** <*WORD*>
telnet vrf<*WORD*> **ipv4 dscp** <>	
telnet vrf <*WORD*> **ipv4 server max-servers** <>	
telnet vrf <*WORD*> **ipv6 dscp** <>	
telnet vrf <*WORD*> **ipv6 server max-servers** <>	

Supported Telnet YANG models in optional RPMs

- Cisco-IOS-XR-ipv4-telnet-cfg
- Cisco-IOS-XR-ipv4-telnet-mgmt-cfg
- Cisco-IOS-XR-um-telnet-cfg
- Cisco-IOS-XR-openconfig-system-terminal-ext

Obtain Install Files

You can obtain the install files based on one of the following options that is best suited to your network:

- **Base ISO and Optional RPMs:** You can upgrade the software through the standard method where you install the ISO followed by the required RPMs.
- **Golden ISO:** You can build a customized golden ISO (GISO) image with the base ISO and the required RPMs to automatically upgrade the software.

Standard ISO and RPMs

Standard ISO and RPMs are base components in the deployment, installation, and management of software systems in a Linux-based environment. They ensure consistency and reliability across diverse systems in a network by providing standardized formats for packaging, distributing, and installing software.

Download Install Files from Cisco Software Center

Obtain the install files (base ISO and RPMs) for the target release.

Procedure

Step 1 Access the Cisco Software Download page.

For optimum website experience, we recommend any of the following browsers: Google Chrome, Mozilla Firefox or Internet Explorer.

Step 2 Select the following:

- Product Name: 8000 Series Routers
- Product Variant: For example, 8201 Router.
- Software Type: IOS XR Software or IOS XR Software Maintenance Upgrades (SMU).

Step 3 From the left pane, select the release.

For the selected release, the Software Download page displays the downloadable files. For more information, see .

Step 4 Use your Cisco login credentials to download the files.

Golden ISO images

The Golden ISO (GISO) image is a custom-built IOS XR software image that

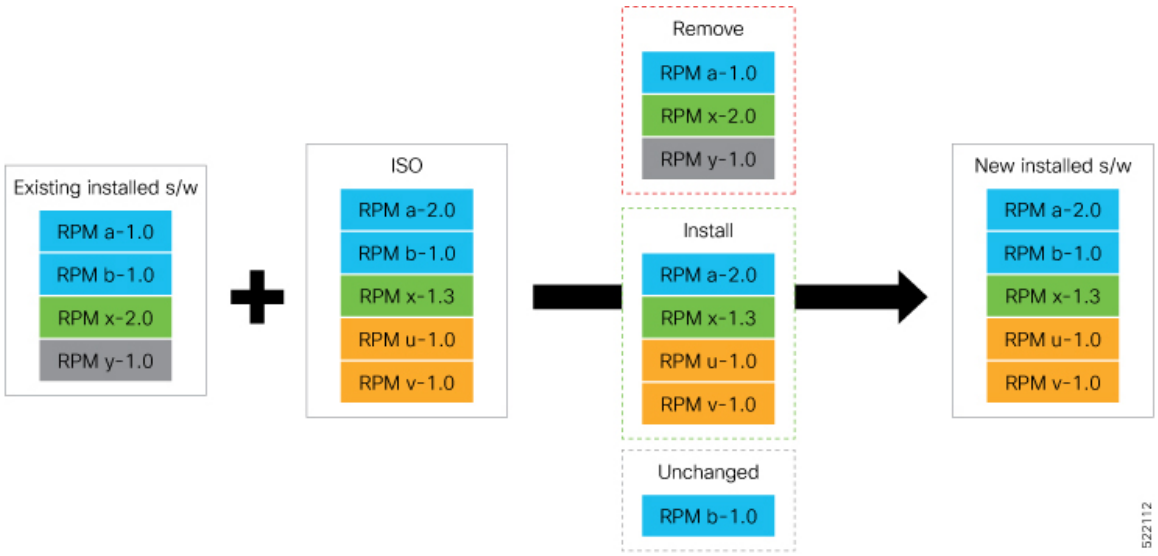
- is based on a given IOS XR version
- contains all mandatory packages
- includes optional packages and SMUs requested by the GISO image owner, and
- enables automated router boot using iPXE or system upgrade to a specified software version in a single operation.

Note When switching between Golden ISOs (GISOs) built on the same base version of IOS-XR, the **install replace** command automatically apples only the minimum required changes such as package additions, removals, and upgrades required for the transition.

Table 9: Feature History Table

Feature Name	Release Information	Description
Build a Golden ISO image using `gisobuild.py` tool	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM
Build a Golden ISO image using `gisobuild.py` tool	Release 7.5.1	This feature facilitates faster deployment by allowing you to build your GISO image without the need to contact Cisco support. You can now select the install files, add your RPMs, repackage them as a custom image, and install the image. In previous releases, you were required to contact Cisco to have your GISO built.

Figure 10: Golden ISO build process



Build a Golden ISO image

This section provides the options that help you build a GISO image.

Procedure

Step 1 Build a GISO image using one of the following options:

- **Option 1:** Contact Cisco Support to build the GISO image with the set of packages based on your requirement
- **Option 2:** Use the `gisobuild.py` tool to build the GISO. For information about the tool and how to use it, see https://github.com/ios-xr/gisobuild.

Step 2 Execute this command to install the GISO image built from previous step similar to installing any other ISO image.

Example:

```
Router# install replace /harddisk:/ncs540l-giso-x64.iso
```

Create Repository to Access Install Files

A **Repository** is a directory where the ISO, RPMs, and their metadata are downloaded. The package manager uses this repository to query the packages.

The repository can either be created locally on the router, or on a remote location that can be accessed through FTP, HTTP, or HTTPS. In a repository, you can create directories based on different Cisco IOS XR platforms, releases or both. You can create and use multiple repositories. The files to be installed can saved in the local repository, remote repository or a combination of both.

Note The Golden ISO (GISO) method does not require you to create a repository. However, you can still install the GISO from a remote repository.

Important Each package is named based on its name, version, software release, and architecture. Hence, any packages that have these attributes in common and differ only by platform are indistinguishable. We recommend that you create different repositories for different platforms and releases.

Create Remote Repository

We recommend that you create an external remote repository that acts as a central repository to be used across devices. This eliminates the need to copy files for future updates to each router individually. It also serves as a single source when new RPMs (bug fixes, packages, updates) are made available.

The remote repository is available only through the Management Ethernet interface of the router. The server hosting the external repository must be able to reach the router using the address of the loopback interface in the VRF. If a VRF has more than one loopback interface, the loopback with the lowest-numbered loopback name is selected. For example, Loopback1 is selected over Loopback2. When using VRF, configure the repository to be reachable using a non-default VRF table. If the repository is reachable through an address in a VRF, specify the name of the VRF.

The following instructions are applicable to Linux distribution systems.

Procedure

Step 1 Create a directory on the server and copy the ISO and all RPMs. For example, name the directory as `remote-repo`. The router must be able to access this directory through FTP, HTTP or HTTPS protocol.

Step 2 Extract the files if the RPM files are archived (.tar format) or compressed (.tgz or .gz format). The files hierarchically arrange in sub directories under the main directory.

Step 3 Convert the directory to a repository using `createrepo` utility on the Linux server. This action creates a directory named *repodata* with the metadata of all the RPMs.

Example:

```
[node]$createrepo --database /var/www/html/
Saving Primary metadata
Saving file lists metadata
Saving other metadata
Generating sqlite DBs
Sqlite DBs complete

[node]$cd /var/www/html/
[node]$ls repodata
```

Note
If you add new packages to the repository, change or remove packages from the repository, you must run the `createrepo` command again to update the metadata. This ensures that the package manager chooses the correct packages.

Step 4 Configure the remote repository on the router.

Example:

For HTTP protocol:

```
Router#config
Router(config)#install repository remote-repo url http://10.194.88.104/<directory-with-rpms>
Router(config)#commit
Thu 02 2022 UTC: config[67542]: Configuration committed by user 'cisco'.
Router(config)#end
```

where:

- `remote-repo` is the repository name.
- `http://10.194.88.104/<directory-with-rpms>` is the HTTP repository URL. Similarly, you can configure FTP or HTTPS repository URL.

Step 5 Verify connectivity to the server and check the contents of the repository.

Example:

```
Router#show install available
Trying to access repositories...
Package        Architecture           Version           Repository
xr-8000-core    x86_64                7.8.1             remote-repo
xr-core         x86_64                7.8.1             remote-repo
```

Only the top-level packages that are available in the repository and not part of the active system are displayed. The contents of the repository are displayed only when the configured repository is valid and the RPMs with the updated metadata are present.

System logs record useful information during the creation of the repository. Check the logs to verify that the repository is valid.

Create Local Repository on the Router

The router can also serve as a repository to host the RPMs. However, you must be a `root-lr` user with access to the router shell. Using a local repository removes the need to set up an external server for software installation. In this method, the image files are copied directly to the router and used to create a repository locally.

Note We do not recommend creating a local repository if you are upgrading multiple routers.

Procedure

Step 1 Create a new directory locally on the router's /harddisk:. For example, name the directory as `new-repo`.

Step 2 Copy the required RPMs and ISO files (using `copy` or `scp` command) to the local directory on the router.

Step 3 Access the shell of the router and untar the RPMs.

Example:

```
Router#run
[node:~]$cd new_repo
[node:~]$tar -xvzf <rpm-name>.tgz
```

Step 4 Exit from the shell.

Step 5 Configure the local repository.

Example:

```
Router#config
Router(config)#install repository local-repo url file:///harddisk:/local_repo
Router(config)#commit
Thu 02 2022 UTC: config[67542]: Configuration committed by user 'cisco'.
Router(config)#end
```

where:

- `new-repo` is the repository name.
- `file:///harddisk:/local_repo` is the local repository URL.

Step 6 Check the contents of the repository.

Example:

```
Router#show install available
Trying to access repositories...
Package        Architecture           Version           Repository
xr-8000-core    x86_64                7.8.1             local-repo
xr-core         x86_64                7.8.1             local-repo
```

Only the top-level packages that are available in the repository and not part of the active system are displayed. The contents of the repository are displayed only when the configured repository is valid and the RPMs with the updated metadata are present.

What to do next

The pre-upgrade tasks are complete. Your router is now ready to be upgraded.

Disk encryption

Disk encryption is a data security mechanism that

- protects sensitive customer information by encrypting data on the disk
- ensures data security if a router leaves the customer premises, preventing unauthorized access to the data stored on the disks, and
- adheres to compliance requirements, including PSB SEC-CRY-ALWAYS-2 and PSB SEC-DAT-CLNSTATE-2.

Table 10: Feature History Table

Feature Name	Release Information	Feature Description
Disk encryption	Release 25.4.1	Introduced in this release on: Fixed Systems (8010 [ASIC: A100])(select variants only*) *This feature is supported on the Cisco 8011-32Y8L2H2FH routers.
Disk encryption	Release 25.2.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q100, Q200, P100], 8700 [ASIC: P100, K100], 8010 [ASIC: A100]); Centralized Systems (8600 [ASIC:Q200]); Modular Systems (8800 [LC ASIC: Q100, Q200, P100]) To ensure data security even when a router leaves the customer's premises, we now provide Linux Unified Key Setup (LUKS) with AES-XTS encryption, robust encryption mechanisms, with keys securely stored in the hardware Trusted Authentication Module (TAM). This encryption capability, enabled by default across all router nodes and immutable, operates seamlessly in the background, allowing you to interact with the file system without impacting your experience, ensuring continuous and reliable data protection. The disk encryption feature impacts the IOS XR software upgrade process. For more information, refer the Cisco 8000 IOS XR Software - Upgrade/Downgrade docs for IOS XR Release 25.2.1, in the Cisco Software Download portal.

How disk encryption works

Disk encryption ensures the security of customer data by encrypting specific partitions, safeguarding against unauthorized access, and providing secure data deletion mechanisms. This process is designed to be seamless and does not require user intervention.

Summary

The key components and features of the disk encryption process are:

- Encryption: Disk encryption is applied to all nodes on the Cisco 8000 routers by default and cannot be disabled. Disk encryption is performed at the partition level, targeting all partitions that may contain customer data. Exceptions include the EFI, boot, and disaster recovery partitions, which remain unencrypted to allow BIOS access.
- LUKS implementation: The encryption uses Linux Unified Key Setup (LUKS), leveraging AES-XTS for encryption.
- Key storage: Encryption keys are securely stored in the hardware TAM (Trusted Authentication Module).
- User interaction: The encryption process is not configurable and does not alter how you interact with the filesystem. The accessibility of files depends on whether or not you are logged in to the router. If you are logged in, every data read is automatically decrypted and appears unencrypted to authorized users. However, without logging in, the data remains inaccessible and cannot be decrypted.
- Secure erase and factory reset: The secure erase and factory reset functionalities are enhanced to delete the encryption key, ensuring that data cannot be recovered after these operations.

Restrictions for disk encryption

Disk encryption introduces the following differences in the installation and upgrade experience in routers.

IOS-XR software upgrades from versions earlier than 25.2.1 to version 25.2.1 or later

- Preservation of harddisk:/
  - The router disk partitions requires reformatting for switching between encrypted and unencrypted layouts, which may lead to data loss.
  - Upgrades across the encryption boundary temporarily save up to 1 GB of files in the `/harddisk:/` partition, starting with the smallest. However, the IOS XR configurations in router is completely preserved.
  - Downgrading from an encrypted IOS XR version to an unencrypted version erases all data on the `/harddisk:/` partition.
- Downgrades to .iso containing owner/partner rpms are not supported. Reinstall these rpms after completing the downgrade.
- The docker partition contents are lost during the upgrade, potentially causing application state loss.

IOS-XR software upgrades starting 25.2.1

- If the router reloads during the upgrade but before executing the **install commit** command, it will attempt to boot the target image instead of the original image.
- FPD upgrade
  - The FPDs upgrade during or immediately after bootup instead of before the reload, increasing router unavailability.
  - Interfaces may experience downtime (flapping) for a few minutes after the install completes.
  - Verify the FPD upgrades are complete using the **show FPD status** command.
- The upgrade using install ISO with control over reload timing (**install package replace** command) is not supported. Attempting to use it results in an error.
- Rollbacks are no longer supported for release upgrades. Use the **install replace** command with the previous golden iso instead.

Guidelines for disk encryption

Disk encryption introduces the following differences in the installation and upgrade experience in routers.

IOS-XR software upgrades from versions earlier than 25.2.1 to version 25.2.1 or later

- Preservation of /harddisk:/
  1. Create an archive of `/harddisk:/` contents and copy it to an external location before performing an upgrade or downgrade.
  2. Restore the archive after completing the operation upgrade or downgrade.

3. Move EEM scripts to `/harddisk:/eem` or `/harddisk:/mirror/eem` to preserve them.

- Interaction with other features
  - The contents of `/var/lib/docker` are erased during the upgrade. Reinstall any rpms installed using appmgr after the upgrade.
  - Upgrades to images with owner/partner rpms may not guarantee full robustness in case of power outage.
  - Execute **disk-encryption deactivate location all** command to disable the partial disk encryption feature before starting the upgrade.

IOS-XR software upgrades starting 25.2.1

- You must upgrade the router only using the **install replace** command with .iso or golden ISO.
- Execute the **show install upgrade-matrix** command to identify the SMUs required to be included in golden iso or to be installed before the upgrade.

Transactionality for release upgrades and downgrades

Transactionality is a behaviour that facilitates a rollback to the original software

- after a failed upgrade, or
- if the router reloads before you commit the changes.

Table 11: Feature History Table

Feature Name	Release Information	Feature Description
Transactionality for release upgrades and downgrades	Release 25.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q200]); Centralized Systems (8600 [ASIC: Q200]) ; Modular Systems (8800 [LC ASIC: Q200]) This feature ensures reliable rollback and system recovery by enabling a robust transactionality capability to release upgrade and downgrade operations thus allowing to rollback to the original software version if the upgrade to a new software fails. This feature, which was removed in Cisco IOS XR Release 25.2.1 as part of the caveats associated with the disk encryption feature is supported again from this release. This feature is supported only in a reimage operation where both the original image and the target image support transactionality.

The transactionality feature for IOS XR release upgrades and downgrades was removed due to an associated caveat in the disk encryption feature introduced in Cisco IOS XR Release 25.2.1. This feature is supported again starting from Cisco IOS XR Release 25.4.1.

The transactionality feature results in a roll back to the original image if the reimage operation is unsuccessful or the router reloads before the commit thus ensuring operational stability and a better user experience.

Benefits of transactionality

These are some benefits that the transactionality capability for release upgrades and downgrades provides:

- Ensures the system recovers to a stable state on failed release upgrade or downgrades.
- Ensures rollback and software state consistency across redundant RPs and multi-node configurations, including Route Processor Switch Over (RPSO) or Route Processor Fail Over (RPFO).
- Preserves user configuration, logs, and package information to ensure a smooth rollback to the original software state and reduce the risk of data loss.
- Lays the groundwork to extend the transactionality capability on additional platforms as they migrate to supported disk layouts.

Prerequistes for transactionality

The transactionality feature for release upgrades and downgrades should be supported on both the original software and the target software for successful upgrades.

Guidelines for transactionality

These are some guidelines for the transactionality feature for release upgrades and downgrades:

- Confirm that both the original and target software support the transactionality feature.
- Recover the system manually if rollback fails due to partition corruption or disk issues.
- Do not use explicit or manual reimage commands (such as **install replace reimage**) if you require transactional rollback.

Restrictions for transactionality

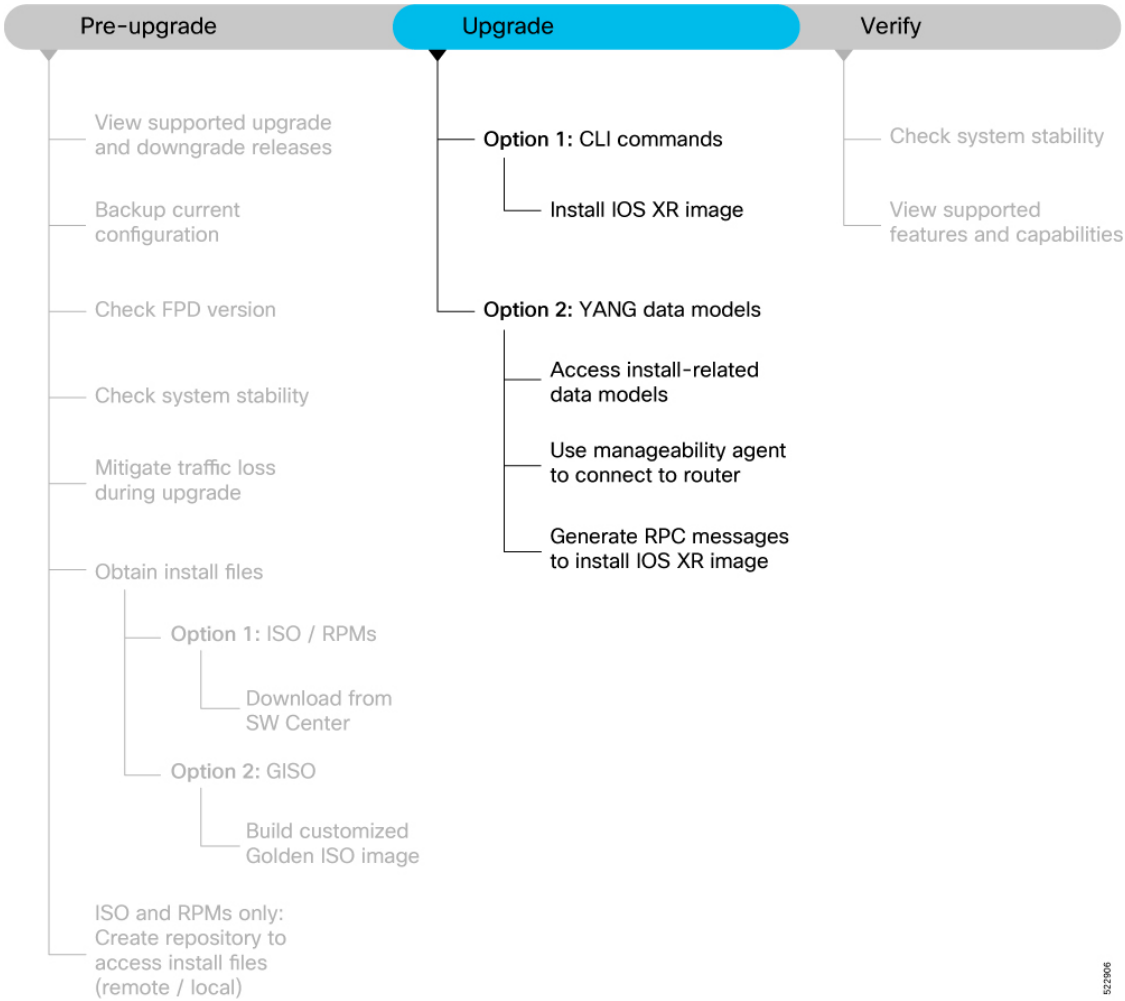
These are some restrictions on the transactionality feature for release upgrades and downgrades:

- Supported only on platforms that have disk encrption.
- The caveat for transactionality associated with the disk encryption feature is removed only if the original software and the target software support the transactionality capability for release upgrades and downgrades.
- Supported only for **install replace** operations.
- Rollback to original image is supported only before an **install commit**.
- **install commit** or **install replace reimage** are the only operations allowed after performing `install replace` for a release upgrade or downgrade and the router restarts.

Upgrade the Software

This section provides information about the processes involved in upgrading the IOS XR software on your Cisco 8000 series routers.

Figure 11: Workflow to Upgrade the Software



The Cisco IOS XR software can be upgraded using one of these methods:

Upgrade Router Using CLI Commands

There are two options to upgrade your Cisco IOS XR software using the Command Line Interface (CLI):

- Base ISO and optional RPMs
- Golden ISO (GISO)

Install IOS XR Image

Install ISO and RPMs

Use this procedure to install the base ISO and optional RPMs.

Before you begin

Ensure you have created a repository locally on the router or on a remote server which is reachable over HTTP, HTTPS or FTP. This repository will be used to copy the required RPMs. Ensure the router can reach the repository server over the Management Ethernet interface. For information about creating the repository to host the RPMs, see Create Repository to Access Install Files, on page 43.

Procedure

Step 1 You can either install from the remote repository or copy the ISO image file to the /harddisk: of the router.

Example:

```
Router#scp root@<ip-address>:/<dir>/8000-x64-release.iso harddisk:
```

Step 2 To verify data integrity, verify the md5 checksum of the copied file with the original MD5 values on CCO.

Example:

```
Router#show md5 file /harddisk:/8000-x64-release.iso
```

Step 3 Install the base image to upgrade the system.

- **Option 1:** Install ISO without control over reload timing.

  ```
  Router#install replace /harddisk:/8000-x64-release.iso
  ```

  The image is installed, the changes are applied through a reload or a restart of the system, and commits the changes. However, you do not have control over the timing of the reload or restart —these occur as soon as the package operation completes and the system is ready.

  If you want to control when your system reloads (management of a network outage), we recommend that you schedule an upgrade window and perform an **install replace**, letting the system reload without intervention.

- **Option 2:** Install ISO with control over reload timing.

  a. Install the image.

  ```
  Router#install package replace /harddisk:/8000-x64-release.iso
  ```

  Note
  This option is now supported only if the ISO or gISO is the same XR release version as the start image and cannot be used for IOS XR release upgrade or downgrade.

  b. Apply the changes.

  ```
  Router#install apply [reload | restart]
  ```

  You can use either the `reload` or `restart` options based on the file that is installed. To determine whether a `reload` or `restart` is required, check the output of **show install request** command. The output indicates the required actions.

Step 4 After the base image is upgraded, install the additional packages. For more information, see Install Additional RPMs and Bug Fixes, on page 152.

If a system fails to boot successfully, or reboots unexpectedly when the package is undergoing a version change, the system is automatically recovered to its old software state.

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.

Install Golden ISO

Table 12: Feature History Table

Feature Name	Release Information	Description
Automatic Bridging of Bug Fix RPMs	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM
Check Integrity of Golden ISO (GISO) Files	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM

Feature Name	Release Information	Description
IOS XR Configuration File in Golden ISO (GISO)	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM
Check Integrity of Golden ISO (GISO) Files	Release 7.5.1	This feature enables an automated check during **install package replace** operations to ensure that the files in GISO have not been corrupted. It does so by calculating the md5sum of the files and comparing it against md5sum value that is contained within the GISO that was calculated when the image was built.
Automatic Bridging of Bug Fix RPMs	Release 7.5.1	In earlier releases, any mandatory bridging bug fixes had to be installed separately *before* a GISO upgrade. In this release, this feature allows mandatory bridging bug fixes to be included within the GISO for installation during the GISO upgrade process. This eliminates the older two-step workflow.

Feature Name	Release Information	Description
IOS XR Configuration File in Golden ISO (GISO)	Release 7.5.1	GISO is a customized image with the standard functional components and additional configuration files. This feature extracts the IOS XR configuration file in GISO and automates the updating of configuration files when the router is reloaded with the new GISO. This feature introduces **iso-config [ignore | replace]** keywords to the **install replace** and **install package replace** commands.

Use this procedure to install the Golden ISO (GISO) that contains the base ISO and a customized list of optional RPMs that you built using the *gisobuild.py* tool. For details, see Golden ISO images, on page 41.

Golden ISO (GISO) upgrades the router to a version that has a predefined list of bug fixes (sometimes also called software maintenance updates) with a single operation.

To update the system to the same release version with a different set of bug fixes:

- Create a GISO with the base version and all the bug fixes you require
- Use the **install replace** or **install package replace** commands to install the GISO.

The GISO can include bridging bug fixes for multiple source releases, and installs only the specific bridging bug fixes 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.
- To meet the prerequisite requirements of a new release version that were not met by the earlier version.

Note The **install replace** command is supported only with GISO, but not with .rpm packages directly.

Procedure

Step 1 Copy the GISO image file to either the /harddisk: of the router or a repository based on your requirement.

Example:

In this example, the image is copied to the /harddisk: of the router.

```
Router#scp root@<ip-address>:/auto/tftp-test/8000-x64-release.iso harddisk:
```

Step 2 Install the GISO.

- **Option 1:** Install GISO without control over reload timing.

a. Install GISO to upgrade to a new release, add or remove bugfixes or optional packages.

```
Router#install replace source-location/giso-name.iso
```

The *source-location* can be one of the following locations based on step 1.

- Local path to the GISO—files located in or under `/var/xr/disk1/`, `/harddisk:/` or `/misc/disk1/`
- Remote repository—`ftp://<server>[;<vrf>]/<remote_path>` or `http://<server>[;<vrf>]/<remote_path>`

This command runs the replace operation and applies the new version via router restart or reload, whichever is least impactful, given the change. For example, if you have a GISO that is the same as your base image except one bugfix, and that bugfix can be applied by process restart, the command will install the bugfix and apply by restart, no router reload occurs. However, you do not have control over the timing of the reload or restart—these operations occur as soon as the packaging is complete and the system is ready. If you want to control the timing of system reloads, we recommend that you schedule an upgrade window and run the **install replace** command, allowing the system to reload without manual intervention or network impact.

b. [Optional] Specify **reload** keyword to force reload for all operations. This may be useful if you want a reliable flow.

c. [Optional] Specify **commit** keyword for the install, apply and commit operations to be performed without user intervention.

- **Option 2:** Install GISO with control over reload timing.

a. Install GISO to upgrade to a new release, add or remove bugfixes or optional packages. The functionality is similar to **install replace** command, except that the staging of packaging changes is performed using this command.

```
Router#install package replace source-location/giso-name.iso
```

The **install package replace** command does not apply the changes.

b. Apply the changes.

```
Router#install apply [reload | restart]
```

You can use either the `reload` or `restart` options based on the change that is installed. You can only apply the changes by restarting the software if the difference between the GISO being installed and the running image is minimal such as bugfixes or package updates.

To determine whether a `reload` or `restart` is required, check the output of **show install request** command. The output indicates the required actions.

Note
A GISO label is a string that identifies a GISO. Any install operation, such as adding or removing a package or modifying the software image (replace or package replace) will change the custom label to a system-generated default label. For example:

```
Router#show install active summary
Build Information:
Built By     : user1
Built On     : Thu Feb 02 09:47:56 UTC 2023
Build Host   : host
Workspace    : /ws
Version      : 7.8.1
```

```
Label        : GISO1
...
```

In this example, the software image is modified to remove the CDP package.

```
Router#install package remove xr-cdp

Install remove operation 39.1.1 has started
Install operation will continue in the background
...
Packaging operation 39.1.1: 'install package remove xr-cdp' completed without error
```

Apply the changes.

```
Router#install apply
Thu Feb 02 11:13:09.015
Once the packaging dependencies have been determined, the install operation may have to reload the
system.
If you want more control of the operation, then explicitly use 'install apply restart' or 'install
apply reload' as
reported by 'show install request'.
Continue? [yes/no]:[yes] yes
RP/0/RP0/CPU0:Feb 02 11:13:12.771 : instorch[404]: %INSTALL-6-ACTION_BEGIN : Apply by restart 39.1
started
Install apply operation 39.1 has started
Install operation will continue in the background
```

View the software version.

```
Router#show version
Build Information:
Built By     : user1
Built On     : Thu Feb 02 10:06:56 UTC 2023
Build Host   : host
Workspace    : /ws
Version      : 7.8.1
Label        : 7.8.1
```

The `GISO1` custom label is replaced with the label `7.8.1` generated by the system.

Upgrade Router Using YANG Data Models

Data models are a programmatic way of configuring and collecting operational data of a network device. They replace the process of manual configuration and can be used to automate configuration tasks across heterogeneous devices in a network.

Access Install-related Data Models

You can use YANG data models to install and upgrade the router. The data models are packaged with the release image in the `/pkg/yang` directory.

Procedure

Step 1 Navigate to the directory in the release image where the YANG data models are available.

Example:

```
Router#run
[node_RP0_CPU0:~]$cd /pkg/yang
```

Step 2 View the list of install-related data models on your router.

Example:

```
node0_RP0_CPU0:/pkg/yang]$ls -ltr *install*
-rw-r--r--.    1    root    root    8646    Jul    2    01:59    Cisco-IOS-XR-install-act.yang
-rw-r--r--.    1    root    root    7267    Jul    2    01:59    Cisco-IOS-XR-install-search-act.yang
-rw-r--r--.    1    root    root    10664   Jul    2    01:59    Cisco-IOS-XR-install-augmented-act.yang
-rw-r--r--.    1    root    root    2511    Jul    2    02:00    Cisco-IOS-XR-um-install-cfg.yang
-rw-r--r--.    1    root    root    2270    Jul    2    02:04    Cisco-IOS-XR-install-cfg.yang
-rw-r--r--.    1    root    root    6222    Jul    2    02:04    Cisco-IOS-XR-install-oper.yang
-rw-r--r--.    1    root    root    14009   Jul    2    02:04
Cisco-IOS-XR-install-augmented-oper.yang
```

The following table describes the function of the install-related data models:

Date Model	Description
Cisco-IOS-XR-um-install-cfg	Unified data model that contains a collection of YANG definitions for Cisco IOS XR install package configuration, and augments the modules with configuration data.
Cisco-IOS-XR-install-oper	Operational data model to view details that are related to basic package information, active and committed packages, and fixes.
Cisco-IOS-XR-install-cfg	Configuration data model to specify the location of the install source.
Cisco-IOS-XR-install-act	Action model to perform basic install operations and software upgrade.
Cisco-IOS-XR-install-search-act	Action model that contains a collection of YANG definitions for install actions related to searching for package information.
Cisco-IOS-XR-install-augmented-oper	Augmented operational model that displays information aboutpackaging, atomic changes, and history of the install operation on the router.
Cisco-IOS-XR-install-augmented-act	Action model to perform flexible install operations, includingcontrolling the exact timing of system reloads and rolling back to a previous commit.
Cisco-IOS-XR-shellutil-copy-act	Action model to copy files on the router from a source location.

You can also access the supported data models to install Cisco IOS XR software from the Github repository.

Use Manageability Agent to Connect to Router

Use a manageability agent like NETCONF or gRPC to connect and communicate with the router. You can send Remote Procedure Calls (RPC) requests to configure or retrieve operational data from the router. The router processes the request and responds to the request through an RPC response. You use the RPCs to send requests to install the software by populating the relevant parameters of a container and leaf in the data model.

For more information about understanding the data model structure and using data models, see the *Programmability Configuration Guide for Cisco 8000 Series Routers* .

Generate RPC Messages to Install IOS XR Image

Before you begin

Not all software versions are supported as the target upgrade software version. You must review the supported upgrade and downgrade paths, hardware or software limitations, and bridging SMUs required for the version. For more information about checking the release support between the current and target versions, see Supported upgrade and downgrade releases, on page 30.

Procedure

Step 1 Use the `install-replace` RPC on the `Cisco-IOS-XR-install-act.yang` data model to upgrade the router(s). Configure the values of the `source-type`, `source`, and `file` parameters.

Example:

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <install-replace xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-install-act>
    <source>/harddisk:</source>
    <file>8000-x64.iso</file>
    <source-type>local</source-type>
    <commit>false</commit>
</install-replace>
</rpc>
```

To specify any additional parameters, see the Cisco-IOS-XR-install-act.yang data model. If not specified, the default values of these parameters are used.

Step 2 Use the `install-commit` RPC on the `Cisco-IOS-XR-install-act.yang` data model to commit the upgrade configuration.

Example:

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <install-commit xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-install-act />
</rpc>
```

You can also specify two optional parameters using the `install-commit` RPC to enforce the commit operation and store the specified description:

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <install-commit xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-install-act>
    <force>false</force>
    <description>Description to be stored</description>
  <install-commit>
</rpc>
```

Step 3 Use the `get` RPC on the `Cisco-IOS-XR-install-oper.yang` data model to verify whether the software upgrade is successful. Edit the `install` parameter and send an RPC request from the client to the router.

Example:

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <get>
   <filter>
    <install xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-install-oper">
```

```
     <request/>
    </install>
   </filter>
  </get>
</rpc>
```

View the RPC response received from the router to check the state of the install operation.

```
<?xml version="1.0"?>
  <rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <data>
    <install xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-install-oper">
     <request>
      <request>install commit</request>
      <state>success</state>
     <timestamp>2022-06-27 T02:52:07Z</timestamp>
    <operation-id>26</operation-id>
   </request>
  </install>
```

Here, the state of the install operation in the RPC response indicates that the software and the RPMs are upgraded successfully.

What to do next

Perform prelimiary checks to verify that the router is upgraded successfully.

Upgrade QDD Optical Modules

The QDD optics firmware file needs to be copied to the router manually. Contact Cisco Support to check the QDD firmware version, IOS XR release compatibility, and to obtain the QDD optics firmware file.

Starting from Cisco IOS XR Release 7.5.2, you can upgrade the Field-Programmable Device (FPD) for QDD optical modules.

Limitation: When ports share a common management interface, IOS XR serializes the firmware upgrade. Serializing and deserializing may delay the upgrade process.

Procedure

Step 1 Copy the QDD firmware file to the harddisk: location.

Example:

```
scp user@10.1.1.1:/home/user/filename harddisk:/
```

When you are using VRF, use the following sample command:

```
scp user@10.1.1.1:/home/user/cl1.bin vrf MGMT harddisk:/

Tue Jan 25 02:57:22.762 UTC
Connecting to 10.1.1.1...
Password:
  Transferred 1484800 Bytes
  1484800 bytes copied in 0 sec (22161194)bytes/sec
```

```
RP/0/RP0/CPU0:8808#dir harddisk:/cl1.bin
Tue Jan 25 03:00:47.835 UTC

Directory of harddisk:/cl1.bin
35 -rw-r--r--. 1 1484800 Jan 25 02:57 dp04qsdd_dp04sfp8_161_10_01.ackit

53461500 kbytes total (42983204 kbytes free)
```

When you are not using VRF, remove the `vrf MGMT` command:

```
scp user@10.1.1.1:/home/user/cl1.bin harddisk:/
```

Step 2 Upgrade the FPD for QDD optical modules.

Example:

Multiple port upgrade:

```
Router#upgrade optics port 0,1,2,3,4 filename /harddisk:/cl1.bin location 0/1/CPU0
```

Single port upgrade:

```
Router#upgrade optics port 0 filename /harddisk:/cl1.bin location 0/1/CPU0
```

Step 3 Check the firmware upgrade progress.

Example:

```
Router#show optics firmware upgrade port 0,1,1,2,3,4 location 0/1/CPU0
```

HTTPS copy and install operation requirements

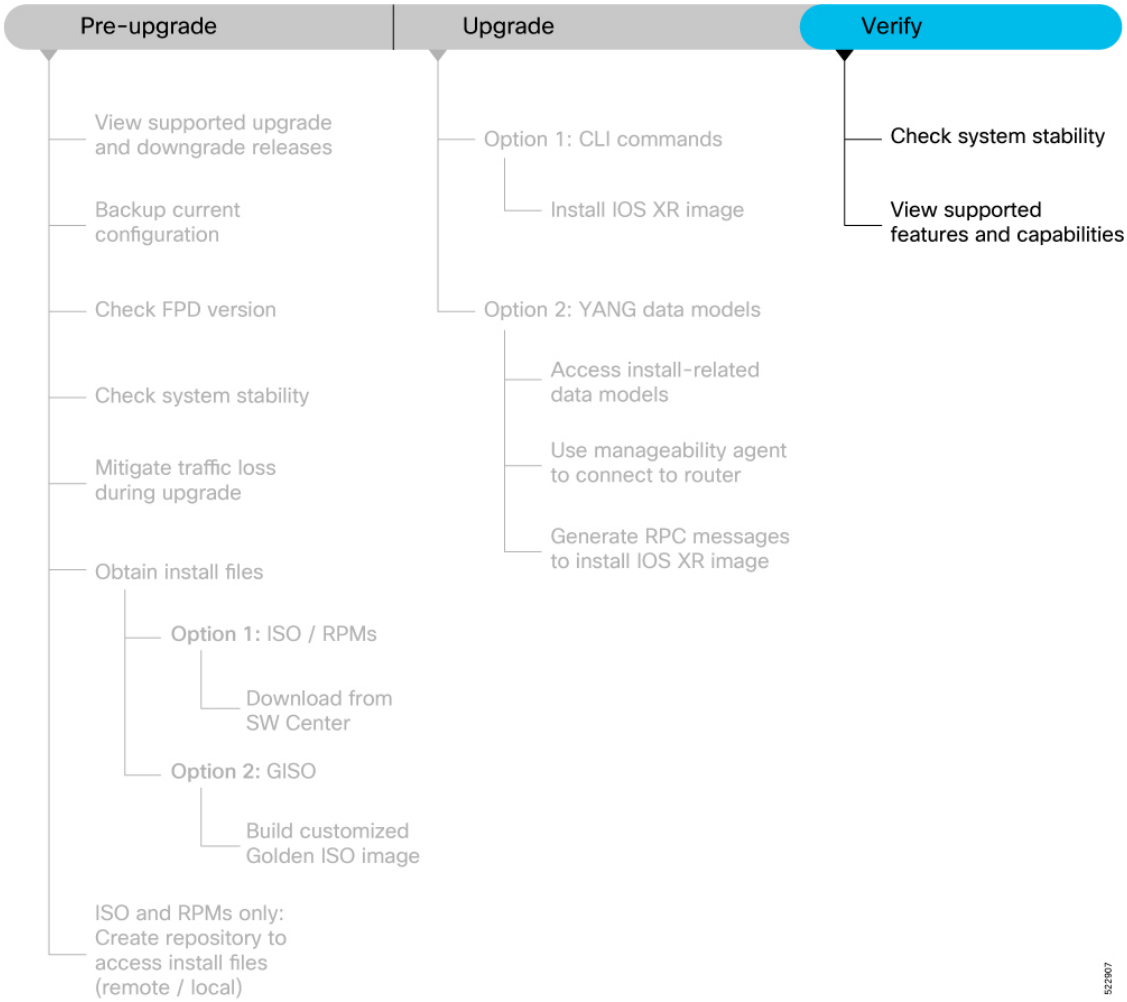
Starting with IOS XR Release 26.3.1, HTTPS copy and install operations require strict TLS session shutdown. If the remote HTTPS server closes the TCP connection without sending a TLS `close_notify` alert, the operation fails.

For a successful HTTPS copy and install operation, configure or update the remote HTTPS server to send a proper TLS `close_notify` alert prior to closing the connection.

Verify the Software Upgrade

This section provides information about the processes involved in verifying the upgraded software on your Cisco 8000 series routers.

Figure 12: Workflow to Verify the Software Upgrade



This section contains the following topics:

Check System Stability

System stability checks are essential to measure the efficiency and ability of an upgrade to function over an extended period.

At the EXEC prompt, execute the following commands to assess basic system stability checks before and after the software upgrade.

Command	Reason	Workaround
show platform	Verify that all nodes are in `IOS XR RUN/OPERATIONAL` state	NA
show redundancy	Verify that a standby RP is available, and the system is in `NSR-ready` state	NA

Command	Reason	Workaround
show install active summary	Verify that the proper set of packages are active	NA
show install committed summary	Verify that the proper set of committed packages are same as active	Execute 'install commit' command
clear configuration inconsistency	Verify/fix configuration file system	NA
show hw-module fpd	Ensure all the FPD versions status are CURRENT	Execute `upgrade hw-module fpd` command
show media	Display the current state of the disk storage media	To free up space, remove older .iso image files and bug fix .tar files.
show inventory	Show chassis inventory information	NA

View supported features and capabilities

Table 13: Feature History Table

Feature Name	Release Information	Description
View Supported Features and Capabilities	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.

Note The **show features** command is not supported from Cisco IOS XR Release 7.9.1 and later.

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) is supported.

```
Router#show features
Fri Sep 1 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 Sep 1 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 Sep 1 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 --------------------------
```

Optics firmware upgrades

An optics firmware upgrade is a software maintenance procedure that

- maintains compatibility between pluggable optics and platform software
- improves performance by providing functional enhancements and defect fixes, and
- enables updates to supported pluggable options using CLI commands executed on the device CPU.

How optics firmware upgrades work on Cisco 8000 Series routers

The firmware upgrade process transfers the firmware image to the router and programs each optic through the platform optics driver. The upgrade operation is performed per optic port.

- During the upgrade, the optic may temporarily stop forwarding traffic. Schedule the operation during a maintenance window when service interruption is acceptable.
- The upgrade duration depends on the firmware image size and may take approximately 10 minutes per MB of firmware image size per optic.
- Multiple optics can be upgraded in parallel, provided they are connected through separate hardware buses and there are sufficient system resources for concurrent operations.

Upgrade optics firmware

Ensure your optic operates with the latest supported features and device compatibility by updating its firmware.

Upgrading the firmware for optic modules may resolve compatibility issues, enable new capabilities, or enhance network reliability. This task applies to Cisco 8000 Series routers equipped with supported optics.

Before you begin

- Ensure that the controller optics is up and running.
- Have the required firmware binary file on the router file system and available to the CPU that manages the optic.

Follow these steps to upgrade optics firmware:

Procedure

Step 1 Copy the firmware image to the router.

Example:

```
Router# scp imagecopy@1.2.15.6:/ws/user-sjc/test-file /misc/disk1/test-file
Tue Apr 28 16:26:21.020 UTC
RP/0/RP0/CPU0:Apr 28 16:26:21.320 UTC: ssh_syslog_proxy[1213]: %SECURITY-SSHD_SYSLOG_PRX-6-INFO_GENERAL
 : ssh_csm[37984]: Warning: Permanently added '1.2.15.6' (ECDSA) to the list of known hosts.
This is a Cisco managed device to be used only for authorized purposes.
Your use is monitored for security, asset protection, and policy compliance.

Password:
RP/0/RP0/CPU0:Apr 28 16:26:24.502 UTC: ssh_syslog_proxy[1213]: %SECURITY-SSHD_SYSLOG_PRX-6-INFO_GENERAL
 : ssh_csm[37984]: SSH Client successfully established session as imagecopy to 1.2.15.6
test-file
                          100%    0     0.0KB/s   00:00
RP/0/RP0/CPU0:Apr 28 16:26:24.839 UTC: ssh_syslog_proxy[1213]: %SECURITY-SSHD_SYSLOG_PRX-6-INFO_GENERAL
 : ssh_csm[37984]: SSH Client terminated session as imagecopy to 1.2.15.6

Router# dir /misc/disk1/test-file
Tue Apr 28 16:26:33.223 UTC

Directory of /misc/disk1/test-file
37 -rw-r--r--. 1 0 Apr 28 16:26 /misc/disk1/test-file

41968752 kbytes total (31013936 kbytes free)
```

Step 2 Identify the optic interface and confirm detection.

Example:

```
Router# show controllers optics 0/0/0/2 | begin Vendor

Wed Jul  2 06:16:01.034 UTC
 Transceiver Vendor Details

         Form Factor             : OSFP
         Optics type             : OSFP 800G DR8
         Name                    : CISCO-CISCO
         OUI Number              : 00.00.0c
         Part Number             : 10-3687-0100-A01
         Rev Number              : 03
         Serial Number           : CGC29360ENK
         PID                     : OSFP-800G-DR8
         VID                     : V01
         Hardware Version        : 1.1
         Firmware Version        : Major.Minor.Build
                 Active          : 3.8.75
                 Inactive        : 3.8.75
         Date Code(yy/mm/dd)     : 25/09/04
```

Review *Transceiver Vendor Details* to confirm the optic and its attributes.

Step 3 Upgrade the firmware for the optic.

Example:

```
Router# upgrade optics port 2 filename
/misc/disk1/OSFP-800G-DR8_CISCO-CISCO_10-3687-0100-A01_HW_1.1_FW_3.8.bin location 0/RP0/CPU0

Wed Jul  2 06:10:22.975 UTC

Ctrl: 0 Port 2
==================
FW Upgrade Trigger Success

Router#
```

Firmware Upgrade Trigger Success appears.

Step 4 Monitor the upgrade process.

Example:

```
Router# show optics firmware upgrade port 2 location 0/RP0/CPU0

Wed Jul  2 06:10:36.822 UTC

Ctrl: 0 Port 2
==================
download::[0], commit::[0]

Router#
Router# show optics firmware upgrade port 2 location 0/RP0/CPU0
Wed Jul  2 06:10:48.240 UTC

Ctrl: 0 Port 2
==================
download::[3], commit::[0
```

Review stages indicated by*download* and *commit* statuses.

Step 5 Verify the upgrade result from system logs.

Example:

```
Router# show logging | include UPGRADE
```

```
Wed Jul  2 06:15:38.964 UTC
RP/0/RP0/CPU0:Jul  2 00:18:24.199 UTC: fpd-serv[241]: %PKT_INFRA-FM-3-FAULT_MAJOR : ALARM_MAJOR
:FPD-NEED-UPGRADE :DECLARE :0/RP0/CPU0:
RP/0/RP0/CPU0:Jul  2 06:10:43.014 UTC: optics_driver[391]: %L2-OPTICS-6-QDD_FW_UPGRADE_STARTED : QDD
 Firmware upgrade started on Optics0/0/0/2
RP/0/RP0/CPU0:Jul  2 06:14:53.849 UTC: optics_driver[391]: %L2-OPTICS-2-QDD_FW_UPGRADE_DONE : QDD
Firmware upgrade completed successfully on Optics0/0/0/2
Router#
```

Look for log entries such as *QDD Firmware upgrade started* and *QDD Firmware upgrade completed successfully*.

The optic module firmware is successfully upgraded. Log messages confirming the start and completion of the upgrade may appear up to one minute after the upgrade commad is issued.

What to do next

Review system logs for confirmation. If issues occur, re-run the upgrade or consult Cisco support resources.

Deprecated commands

The following table lists deprecated commands along with their respective releases.

Table 14: Deprecated commands

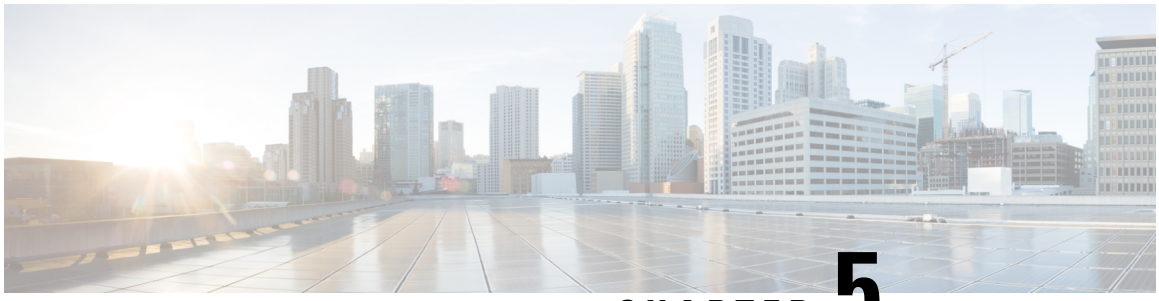
Deprecated command	Release deprecated	Description
show install packages	Release 26.3.1	The **show install packages** family of commands including **show install packages info**, **show install packages search**, and **show install packages provides** are deprecated.
show install fixes deactivate	Release 26.3.1	To remove or deactivate a bug fix, see Deactivation of bug fixes.

Deactivation of bug fixes

To deactivate or remove a bug fix, use one of the following workflows:

- If there was a previous point where the same version of XR was loaded without the bug fix, revert to that point using the **install rollback** command. Use **show install rollback ?** to list the commands you can use to identify a suitable rollback point.
- If a suitable previous gISO is available without the bug fix, install it using the **install replace** command.
- If there is no previous point with the desired set of bug fixes, build a new gISO.

Note If the gISO build fails or the "bad bugfix" is still included, the set of desired fixes is not feasible. In this case, contact Cisco support to obtain a new version of the bad bug fix.



CHAPTER 5

Deploy Router Using Bootz

With the Bootz process, you can securely and seamlessly provision network devices accurately within minutes and without any manual intervention.

Table 15: Feature History Table

Feature	Release Information	Feature Description
Provisioning Using Bootz Process	Release 7.11.1	This feature allows devices in the network to establish a secure connection with the remote Bootz server and authenticate information using a three-step validation process. This process involves validating the network device, the Bootz server, and the onboarding information thereby mitigating security risks and preventing malicious actions during remote provisioning.

Unlike the Secure ZTP process, which relies on vendor-specific definitions for bootstrapping a device, the Bootz process offers a specification that outlines data elements in a vendor-agnostic manner. It also details the necessary operations at turn-up time, integrating them into the boot process.

Also, the bootstrap request in the Bootz process includes the unique identifier or serial number for each node as opposed to the Secure ZTP process where the bootstrap request does not include serial numbers. The Bootz server returns the signed onboarding information with ownership voucher and owner certificate for the requested serial number of the device.

- Supported Bootz versions, on page 72
- Components used in the Bootz Process, on page 72
- Onboard Devices Using Bootz Workflow, on page 73
- Obtain Ownership Voucher, on page 74
- Build bootstrapping data, on page 75
- Bootz workflow commit process for hw-module commands, on page 76
- Provision Bootz Using DHCP Server, on page 76
- BootstrapStream RPC for Bootz process, on page 86
- Immutable bootstrap configurations, on page 89

• Bootz dynamic configuration, on page 92

Supported Bootz versions

This table provides the Bootz versions supported in each release. The Bootz Bootstrap server must be compatible with the respective Bootz version.

Table 16: Bootz Versions

Release	Version with File Path
Release 26.2.1	`openconfig/bootz v0.6.1`
Release 25.4.1	`openconfig/bootz v0.5.0`
Releases 24.3.1, 24.4.1, 25.1.1, 25.2.1, 25.3.1	`openconfig/bootz v0.3.1`
Release 24.2.1	`openconfig/bootz v0.1.1-0.20231106050618-8d6e2559f803`
Release 24.1.1	`openconfig/bootz v0.0.0-20230809153947-e6bc0be82dd8`
Release 7.11.1	`openconfig/bootz v0.1.0`

Components used in the Bootz Process

These components are part of the Bootz process.

- **Onboarding Device (Router):** A router is a Cisco device that you want to provision and connect to your network. Bootz is supported only on platforms that have *Hardware TAM*[1] support.
- **DHCP Server:** The DHCP server provides the URL where the Bootz process can access the bootstrapping information.
- **MASA Server:** You can generate and store the ownership voucher in the MASA server. The MASA server sends the ownership voucher to the Bootz server so that the Bootz process validates the device and establishes device ownership.
- **Bootz Bootstrap Server:** A Bootz Bootstrap server is any gRPC server used as a Bootz bootstrapping data source. For example, Google Proto. The Bootz Bootstrap server is complaint with Openconfig Bootz standards.

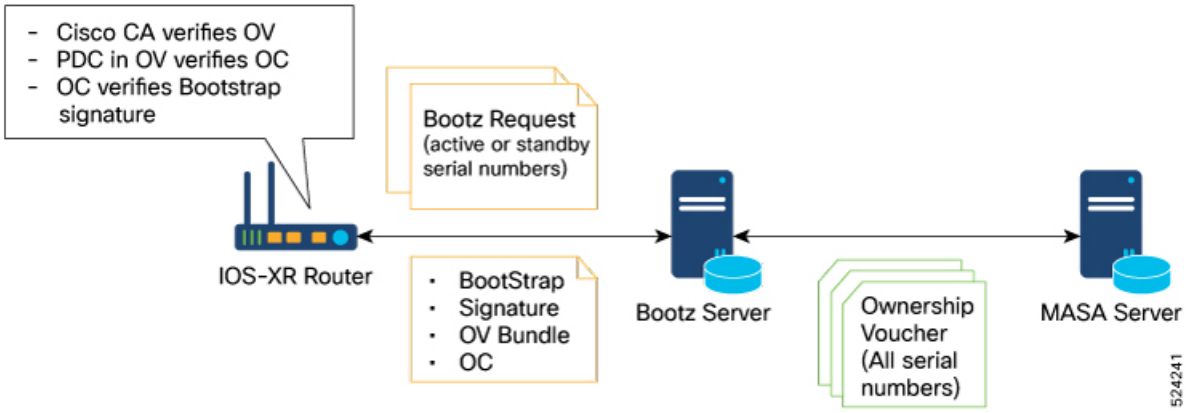
Note Bootz only supports a single name-server. As a result, when the DHCP server has more than one server address configured, Bootz fails to apply the server configuration.

[1] A secure storage device that stores the customer certificates and Cisco's internal secure data like trust anchors, SUDI certificates, secure flags, and other security information.

The Bootz server contains these artifacts:

- **Cisco IOS XR software images:** You can download Cisco images, SMU, and patches from the Cisco Support & Downloads page.
- **Bootstrapping Data:** It is a collection of data that you have created and uploaded to the Bootz server. The router obtains this data from the Bootz server during the provisioning process.

  Starting with IOS XR Release 24.4.1, the bootstrap configuration remains immutable throughout the device's operation. For more information, refer the topic Immutable bootstrap configuration.



Onboard Devices Using Bootz Workflow

The Cisco IOS XR software supports Bootz provisioning capabilities. The Bootz process uses the Google Remote Procedure Call (gRPC) protocol for fetching information from a remote server.

The Bootz workflow performs these validations to onboard the remote devices securely.

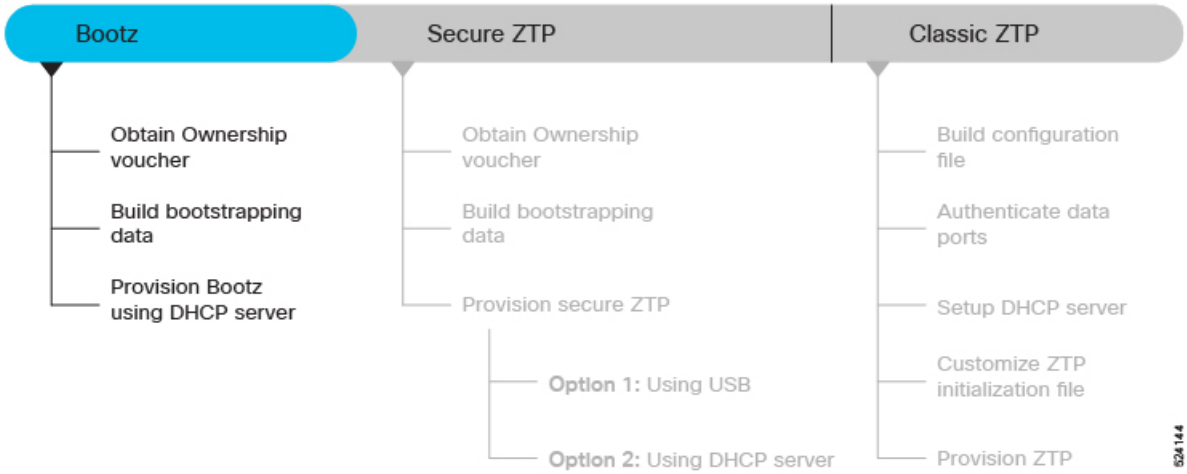
1. **Router Validation**: The Bootz server authenticates the router before providing the bootstrapping data.
2. **Server Validation**: The router in turn validates the Bootz server and ensures that the onboarding is performed for the correct network. Once it is validated, the Bootz server sends the bootstrapping data (for example, a YANG data model) or artifact to the router.
3. **Artifact Validation**: The router validates the bootstrapping data or artifacts received from the Bootz server.

Prior to Cisco IOS XR Release 24.3.1, the Bootz workflow processed the ownership voucher (OV) and onboarded devices only with an active Route Processor (RP). Starting with Cisco IOS XR Release 24.3.1, the Bootz workflow now processes OV information for both active and standby RPs, enabling the onboarding of devices with multiple standby control cards or line cards.

For more information about how the Bootz workflow works for the standby RP, see the Bootz Workflow for Standby RP.

This figure provides the Bootz workflow and the processes involved in the workflow. The sections that follow describe these processes in detail.

Figure 13: Bootz Workflow



Obtain Ownership Voucher

The ownership voucher is used to identify the owner of the device by verifying the owner certificate stored in the device.

Table 17: Feature History Table

Feature	Release Information	Feature Description
Ownership voucher request enhancement	Release 25.3.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q200, P100], 8700 [ASIC: P100, K100]); Modular Systems (8800 [LC ASIC: Q100, Q200, P100]) From this release onwards, when requesting an ownership voucher for fixed systems, you must use the chassis serial numbers in the Ownership Voucher request. Additionally, for both modular and fixed chassis systems, the serial number included in the ownership voucher request must match the SUDI subject name or the certificate on the device. This ensures accurate device identification and owner verification.

How to obtain Ownership Voucher

These steps help you obtain the ownership voucher from Cisco:

1. Contact Cisco Support.
2. Provide these information in your request to Cisco.
    - **Pinned Domain certificate (PDC):** PDC is an X.509 v3 certificate structure that uses Distinguished Encoding Rules (DER). The router uses this certificate to trust a public key infrastructure for verifying a domain certificate supplied to the router separately in the bootstrapping data. This certificate could be an end-entity certificate, including a self signed entity.

- Purchase order details with the serial numbers of the routers.

Sample Request:

```
{
   "expires-on": "2016-10-21T19:31:42Z",
   "assertion": "verified",
   "serial-number": "JADA123456789",
   "idevid-issuer": "base64encodedvalue==",
   "pinned-domain-cert": "base64endvalue==",
   "last-renewal-date": "2017-10-07T19:31:42Z"
}
```

3. Cisco generates the ownership voucher in .vcj format (Example: DCA213140YX.vcj) and sends the voucher in response to your request.

Note

- On fixed systems, use chassis serial number in the Ownership Voucher request.
- On modular systems, use Route Processor serial number in the Ownership Voucher request.

Build bootstrapping data

Bootstrapping data is the configuration data that

- the router retrieves from the Bootz server using gRPC protocols
- configures and sets up the router during its initial boot sequence
- comprises parameters such as gRPC services to be run and the management connectivity setup, and
- is immutable after the initial router boot up, from IOS XR Release 24.4.1 onwards. For more information, see Immutable bootstrap configurations.

Steps to build the bootstrapping data:

1. Create and upload the bootstrapping data to the gRPC server or Bootz bootstrap server.
2. The router sends a bootstrap request with these artifacts to the Bootz server.
   - Serial number of the control card or line card
   - Software image to download and install
   - Bootloader Password for the device
   - Certificate used to validate the bootstrap server
   - Bootstrap server configuration information such as server credentials, path information, authentication information, and certificates

   For the request message format, see the Bootstrap Request Message.

3. The Bootz server returns the listed bootstrapping data in its response to the router. The router receives these data during the provisioning process.
   - **Signed Bootstrap Response:** Each bootstrap response contains the onboarding information for:
     - A single control card or line card for active RP.
     - One or more control cards or line cards for standby RP.

     For the response message format, see the Bootstrap Response Message for a single card.
   - **Owner Certificate:** The owner certificate is installed on the router with your organization's public key. The router uses this public key in the owner certificate to verify the signature in the signed bootstrap response artifact.
   - **Ownership Voucher:** The ownership voucher is used to identify the device owner by verifying the owner certificate stored in the device. Cisco generates and supplies the ownership voucher in response to your request containing the PDC and device serial numbers. For more information, see How to obtain Ownership Voucher.
4. When the router obtains the onboarding information from the Bootz server, the router reports the bootstrapping progress to the Bootz server using the API calls.

Bootz workflow commit process for hw-module commands

The Bootz workflow now requires a two-step commit process for **hw-module** prefixed configuration:

1. Preconfiguration commit: If Bootz's *BootstrapDataResponse* message contains **hw-module** configurations, Bootz executes a pre-configuration stage to commit hw-module prefixed configuration. This stage includes a mandatory device reload prior to the actual device provisioning.
2. Configuration commit: After the device reload, perform the actual configuration commit.

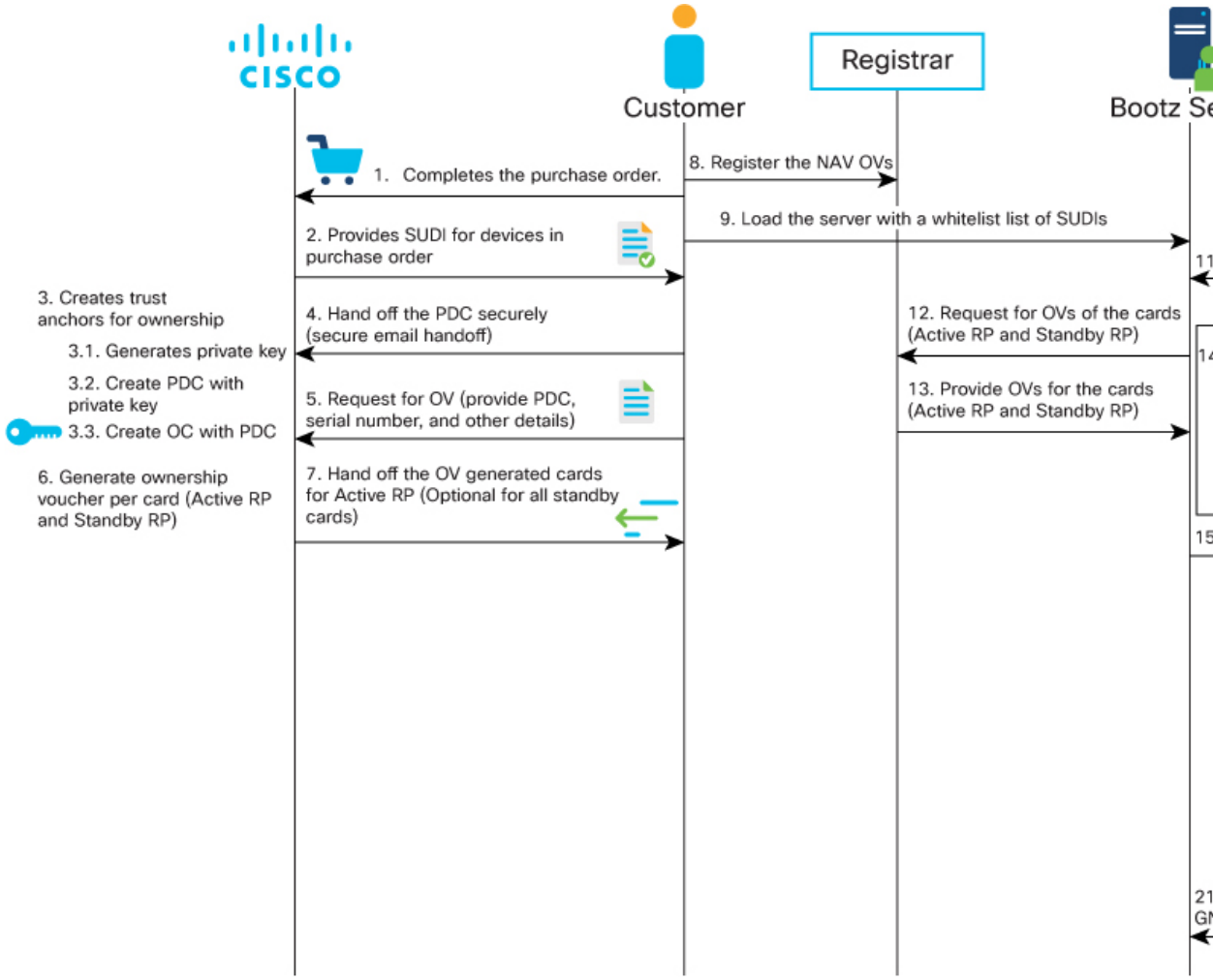
This approach ensures that changes to hardware modules are processed safely and in the correct sequence.

Provision Bootz Using DHCP Server

When you boot the device, the Bootz process initiates automatically on a device without prior configuration. During the process, the router receives the details of the configuration file from the DHCP server.

This figure illustrates the end-to-end sequence of the Bootz process:

Figure 14: End-to-end sequence of the Bootz process



Before you begin

As part of the initial setup for secure ZTP, the network administrator:

- Ensures to enable secure ZTP on the router using the **ztp secure-mode enable** command and reload the router.
- Contacts Cisco Support and follows the steps in How to obtain Ownership Voucher to obtain a voucher from Cisco.

Procedure

Step 1 Upload the listed bootstrapping data to the Bootz server. Refer to your vendor documentation as the upload procedure may vary from server to server.

- Cisco IOS XR software images

Note
Download Cisco images, SMU, and patches from the Cisco Support & Downloads page.

- Serial numbers of the routers to be onboarded
- Owner certificates
- Pinned Domain Certificate (PDC)
- Ownership vouchers

Step 2 Set up the DHCP server to provide the redirect URL to the router:

Before triggering the secure ZTP process, configure the DHCP server so that it provides the location of the IOS-XR image to the router. For information about how to configure the DHCP server, see your DHCP server documentation.

Configure these parameters in the DHCP server:

- `option-code`: Use one of these DHCP SZTP redirect option parameters in the `option-code` setting.
  - OPTION_V4_SZTP_REDIRECT (143): DHCP v4 code for IPv4.
  - OPTION_V6_SZTP_REDIRECT (136): DHCP v6 code for IPv6.
- `option-length`: Provide the option length in octets.
- `bootstrap-servers`: A list of servers. The onboarding device contact these servers for the bootstrapping data.

  `"bootz://<ip-address-or-hostname>[:<port>]<endpoint>"`

Example: `option dhcp6.bootstrap-servers code 136 = text;`

Step 3 Power on the router.

This procedure provides the high-level workflow of the Bootz process:

a. When you boot the device with an IOS-XR image, the secure ZTP process verifies if the secure ZTP mode (`secure-ztp mode`) is enabled. If not enabled, the device boots normally.

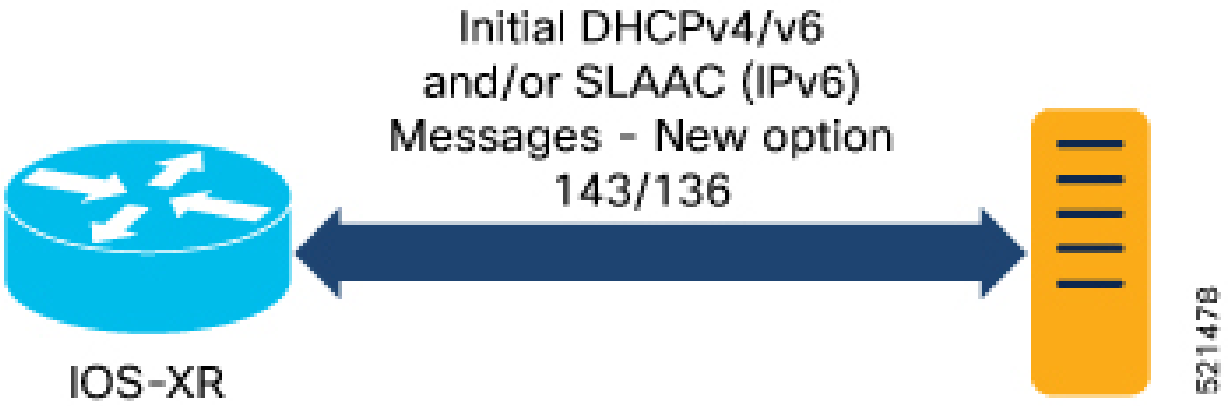
Note
When `secure-ztp mode` is enabled, the ZTP process accepts only the `secure-redirect-URL` and ignores the presence of the boot file name option from the DHCP response.

b. **DHCP discovery**:

1. The router initiates a DHCP request to the DHCP server.
2. The DHCP server responds with a DHCPv4 143 address option (for IPv4 addressing) or a DHCPv6 136 option (for IPv6 addressing).

Note
URLs to access bootstrap servers for further configuration are listed in options 136 and 143.

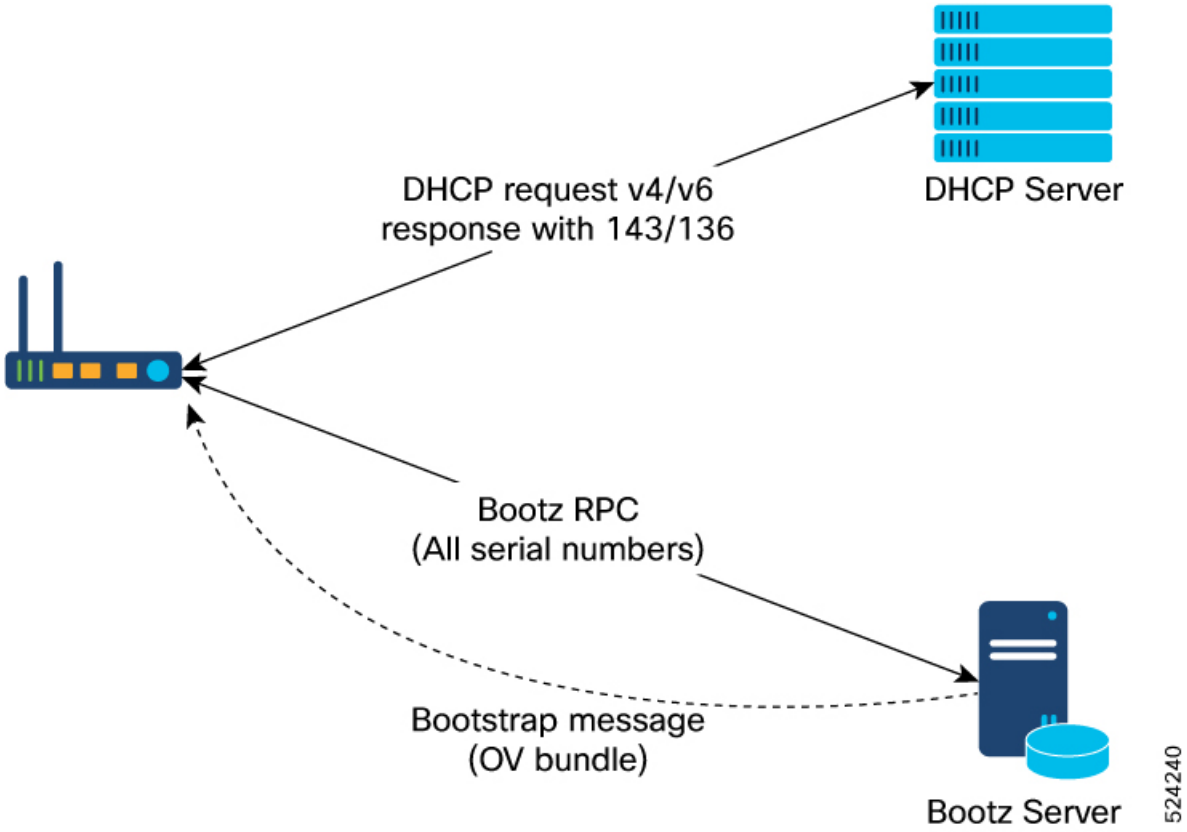
Figure 15: DHCP discovery



c. **Router and Bootz server validation**:

1. After receiving the URL from the DHCP server, the router initiates a gRPC connection to the Bootz server. The Bootz server IP address is obtained from the DHCP response.
2. The Bootz server authenticates the router before it provides the bootstrapping data.
3. After the Bootz server authenticates the router or the onboarding device, the router validates the Bootz server to ensure that the onboarding is performed for the correct network.

   After validating the Bootz server, the router sends the serial number for each control card or line card and other artifacts in its bootstrap request.
4. After its validation, the Bootz server sends the required artifacts along with the bootstrap response data to the router or the onboarding device.



d. Ownership Voucher verification:

The router receives the bootstrap response data that contains owner certificate, ownership voucher for each serial number, and the details of the image upgrade, if any.

Bootstrap response data includes the following:

- Image path
- Image version
- Trust anchor
- Boot configuration
- GNSI artifacts

These artifacts come from the Bootz server as a bootstrap response gRPC message. The router verifies the ownership voucher by validating its signature to one of its preconfigured trust anchors and downloads the image. When the router obtains the onboarding information, it reports the bootstrapping progress to the Bootz server.

If the router has sent the bootstrap request with serial numbers for multiple control cards or line cards, the Bootz server may optionally create an *OV bundle*[2] (.tar file) that contains the ownership voucher for each serial number and return the bootstrap response message with this OV bundle to the router.

[2] The OV bundle contains the ownership voucher for each serial number.

If the Bootz server cannot create an OV bundle, follow the procedure in How the Router Obtains and Processes the OV Information to know how the router obtains the OV information for multiple serial numbers.

e. Artifact Validation:

The router validates the artifacts received from the Bootz server as follows:

1. The device extracts the `pinned-domain-cert` node, an X.509 certificate from the ownership voucher to verify the owner certificate.
2. The device authenticates the owner certificate by performing the X.509 certificate path verification process on the trusted certificate.
3. Finally, the device verifies whether the artifact is signed by the validated owner certificate.

f. Provision the device:

1. The device first processes the boot image information.
2. Executes the script and then onboards the artifacts received from the Bootz server.
3. From 24.4.1 onwards, the device reports the status of the GNSI artifacts to the Bootz server.

g. After the onboarding process is completed, the network device is operational.

Bootz Workflow for Standby RP

Table 18: Feature History Table

Feature	Release Information	Feature Description
Bootz Workflow for Standby RP	Release 24.3.1	

Feature	Release Information	Feature Description
		Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) This feature enables the Bootz workflow to achieve full-system onboarding for devices with both active and standby Route Processors (RPs). In earlier releases, the Bootz workflow only supported onboarding for devices with an active RP. With this enhancement, the Bootz workflow can now detect faulty or tampered standby cards that are inserted dynamically during or after the active RP Bootz process. It does this by verifying the ownership voucher (OV) of the other cards during the initial Bootz process for the active RP. Faulty cards can be shut down to prevent security threats during remote provisioning, ensuring smooth network operation. This feature allows Bootz workflow to validate the standby RP as part of the active RP Bootz process. This feature introduces the **ztp bootz-server** command. This feature modifies the `Cisco-IOS-XR-ztp-cfg.yang`. (see GitHub, YANG Data Models Navigator) *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM

Overview

The Bootz workflow now processes the ownership voucher (OV) for multiple control cards or line cards detected on the standby RP, either before the Bootz process for an active RP starts, during its progress, or after its completion.

With this new feature, the Bootz workflow:

- Allows dynamic insertion or replacement of standby control cards and line cards.
- Processes the ownership voucher (OV) bundle that the Bootz server sends to the router for the standby control cards or line cards.
- Performs the ownership verification of all standby control cards and line cards.

This feature allows you to configure the ZTP Bootz server with the **ztp bootz-server** command to store the server and vendor information received during the initial Bootz process for the active RP.

The router uses this configuration to communicate with the Bootz server and obtain the OV bundle (.tar file) for dynamically inserted or replaced standby control cards or line cards.

Prerequisites

- Configure the Bootz server to return the bootstrap data response message for all the serial numbers of the cards on the device with either the OV bundle or individual ownership voucher for each card.
- Include the **ztp bootz-server** configuration in the server's onboarding information or the vendor configuration information. This configuration is received from the Bootz server during the initial GetBootstrapDataRequest exchange for the active RP.
- Ensure that the routers running the Bootz client can process the OV bundle.

Restrictions

The dynamic Bootz workflow for the standby RP is triggered only if these conditions are met:

- Secure ZTP is enabled on the device.
- The Bootz process for the active RP is completed or not in progress.
- The Bootz server configuration from the initial Bootz process for active RP is available for dynamically inserted standby cards.

Use Cases

These use cases describe different scenarios where the standby RP cards are detected and the OV information is processed accordingly.

Use Case 1 - Standby Card Detected Before Bootz Process for Active RP Starts

If both the active RP and standby RP are detected during the initial boot process before the router communicates with the Bootz server:

- The router sends a bootstrap request to the Bootz server, including the serial numbers for both the active RP and standby RP.

- The Bootz server responds with the OV information for both the active RP and standby RP in its bootstrap response.
- If the response message for the initial bootstrap data request (for active RP) does not include an OV bundle for the standby RP, an additional bootstrap data request is triggered to fetch the OV information for the standby RP.

If there is no OV bundle in the bootstrap response for the initial bootstrap data request (for active RP),

Use Case 2 - Standby Card Detected During Bootz Process for Active RP

If the standby card is detected while the Bootz process for the active RP is in progress, the Bootz process for the standby RP is automatically triggered after the active RP's Bootz process completes.

In this scenario, the router uses the server information received during the initial Bootz process for the active RP, which you have configured using the **ztp bootz-server** command, to:

- Communicate with the Bootz server.
- Send the serial numbers for the dynamically inserted cards in its bootstrap request for standby RP to the Bootz server.
- Obtain the OV bundle (.tar file) from the Bootz server and process the OV for each card with a matching serial number. For more information about the OV bundle, see How the router obtains and processes the OV information.

The Bootz server, in turn, sends the OV information for the standby RP in its bootstrap response.

Use Case 3 - Standby Card Detected After Bootz Process for Active RP Completes

If the standby RP is detected after the initial Bootz process for active RP is completed, the Bootz process is triggered again automatically. The Bootz process is re-triggered if one of the following events occur:

- When a new card is inserted.
- When an existing card is replaced with another card.

Store the server information obtained during the initial Bootz process for active RP. As the secure ZTP workflow is not re-triggered for dynamically inserted standby control cards or line cards, the router uses this stored server information to communicate with the Bootz server and obtain the ownership vouchers for the newly inserted standby control cards or line cards.

Configuration to store server information obtained from the active RP Bootz process:

```
RP/0/RP0/CPU0:ios# config
RP/0/RP0/CPU0:ios(config)#ztp bootz-server ip 1.1.1.1 port 5000 trust-anchor
/misc/disk1/ta.cert
RP/0/RP0/CPU0:ios(config)# commit
```

Once the dynamic Bootz workflow for standby RP is triggered, the router communicates with the Bootz server using the server and vendor configuration information specified in the **ztp bootz-server** command. The Bootz server then sends the OV information for the dynamically inserted standby cards in its bootstrap response.

How the Router Obtains and Processes the OV Information

The router uses one of these methods to process the OV information that it has obtained from the Bootz server.

- **If the Bootz server is configured to send an OV bundle:**

1. The Bootz server sends the OV bundle as a single tar file in the bootstrap response to the router for the RPs.
2. The router running the Bootz client processes the tar file to verify individual ownership voucher for each serial number.
3. The router loads the owner certificate and the ownership voucher on each card with a serial number matching the serial number included in the ownership voucher.

- **If the Bootz server is configured to send individual ownership vouchers rather than an OV bundle, the router:**
  1. Communicates with the Bootz server using the server information from the bootstrap response for active RP.
  2. Sends a new bootstrap request for the standby RP
  3. Obtains the ownership voucher information for the standby RP.

BootstrapStream RPC for Bootz process

Initially, Cisco IOS XR devices relied exclusively on the unary GetBootstrapData RPC for the bootz process. This limited security and flexibility, as TLS 1.3 could only be used on platforms supporting trusted platform module (TPM) or elliptic curve cryptography (ECC)-based SUDI certificates. As a result, secure bootstrapping on many platforms was limited to TLS 1.2, and authentication flexibility were narrowly constrained by hardware-dependent certificate support.

Table 19: Feature History Table

Feature	Release Information	Feature Description
BootstrapStream RPC for Bootz Process	Release 26.2.1	Introduced in this release on: Centralized Systems (8400 [ASIC: K100])(select variants only*) BootstrapStream RPC enhances the security, flexibility, and future-readiness of Bootz process by enabling TLS 1.3-based device bootstrapping in Cisco IOS XR regardless of the hardware limitations in SUDI certificate. With the BootstrapStream RPC, backward compatibility with the existing Bootz infrastructure is preserved. This feature introduces these changes: CLI: A new command, **show ztp information bootz tls**, has been added.

Cisco IOS XR Release 26.2.1 introduces a new BootstrapStream RPC offering secure, bidirectional streaming for device bootstrapping through Bootz. Devices can retrieve bootstrap data over TLS 1.3 without requiring a SUDI certificate during the initial TLS handshake. Mutual authentication is managed using a nonce-based challenge–response mechanism, where the device signs a server-provided nonce with its SUDI private key.

BootstrapStream RPC addresses hardware limitations that previously restricted some platforms to TLS 1.2 for SUDI certificates. When BootstrapStream RPC is not implemented on the Bootz server, the device automatically fall back to legacy GetBootstrapData RPC, maintaining compatibility. BootstrapStream RPC also supports status reporting using the same streaming mechanism previously handled by the legacy ReportStatus RPC.

A new command, **show ztp information bootz tls**, has been introduced for displaying TLS capabilities and security context used by the device during the Bootz bootstrapping process.

Benefits of BootstrapStream RPC

The BootstrapStream RPC feature provides the following key benefits for the Bootz workflow in IOS XR:

- Enables TLS 1.3 bootstrapping across platforms, even without hardware or SUDI certificate support for TLS 1.3 during the TLS handshake.
- Removes dependency on TLS-handshake–based SUDI authentication, shifting device authentication to a nonce-based challenge–response mechanism within the RPC session.
- Improves security by supporting modern cryptographic practices avoiding deprecated RSA-PKCS#1.
- Maintains backward compatibility by automatically falling back to legacy GetBootstrapData RPC when the BootstrapStream RPC is not implemented, ensuring interoperability with existing Bootz servers.

- Requires no user configuration or operational changes. Operates transparently during first boot, preserving the existing ZTP user experience.
- Extends functionality to secure status reporting using the same streaming mechanism with ReportStatus operations.

Limitations of BootstrapStream RPC

The following limitations apply when you use the BootstrapStream RPC feature:

- If the Bootz server implements BootstrapStream and still wants to use the legacy RPC, you must explicitly return `UnImplemented` status code for the BootstrapStream RPC. If the server does not implement BootstrapStream, gRPC returns `Unimplemented` by default.
- While BootstrapStream enables TLS 1.3 transport, all devices still rely on the RSA-based SUDI and PKCS#1-based signature generation for the nonce verification workflow.
- The nonce-based challenge–response mechanism introduces extra message exchanges compared to the unary GetBootstrapData RPC, slightly increasing protocol complexity.
- BootstrapStream is limited to initial bootstrapping and status reporting, and cannot be reused for general configuration or operational data exchange.

How BootstrapStream RPC operations work

Summary

BootstrapStream RPC streamlines the initial configuration and secure ownership verification of Cisco IOS XR devices during their first boot. It ensures only authenticated devices are provisioned, using a series of validated, authenticated exchanges with a Bootz server.

The key components involved in the process are:

- Cisco IOS XR device: Boots in a factory-default state and initiates provisioning.
- Bootz server: Provides bootstrap data and manages authentication challenges during device onboarding.
- SUDI certificate: Ensures device identity and security through cryptographic validation.

Workflow

These stages describe the BootstrapStream RPC workflow:

1. The device boots in a factory-default state and initiates secure ZTP.
2. The device retrieves the Bootz server URI from the DHCP response using Option 143 for IPv4 or Option 136 for IPv6.
3. The device establishes a gRPC connection to the Bootz server using TLS 1.3, without presenting a client certificate during the TLS handshake.
4. The device starts the BootstrapStream RPC by sending a bootstrap request with chassis and control card information along with the device's SUDI certificate.

5. The Bootz server validates the SUDI certificate using the Cisco CA bundle.

   Upon successful validation, the server sends a nonce challenge to the device.

6. The device signs the received nonce using the SUDI private key via the TAM.

   The signed nonce (RSA-PKCS#1 with SHA-256) is sent back to the server.

7. The Bootz server verifies the signed nonce to authenticate the device.

   Once validated, the server sends the bootstrap response, including bootstrap data, ownership voucher and certificate, and a signed response.

8. The device verifies the ownership voucher, ownership certificate, and response signature.

   After successful verification, the device applies the immutable boot configuration and transitions to normal device provisioning.

Result

The BootstrapStream RPC workflow securely provisions Cisco IOS XR devices, automating their configuration while validating device ownership. This enables seamless, secure onboarding into the network for newly deployed devices.

Immutable bootstrap configurations

Immutable bootstrap configuration is the tamper-proof boot configuration of the router that is not affected by subsequent dynamic configuration changes by gRPC Network Management Interface (gNMI).

Table 20: Feature History Table

Feature	Release Information	Feature Description
Immutable bootstrap configurations	Release 25.4.1	Introduced in this release on: Fixed Systems (8700 [ASIC: K100])(select variants only*); *This feature is supported on Cisco 8711-48Z-M routers.

Feature	Release Information	Feature Description
Immutable bootstrap configurations	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100, K100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) You can now ensure your router always boots up in a known good state. This feature prevents unauthorized changes to the router's boot configuration, ensuring that it remains static and tamper-proof. This maintains the integrity and security of the router from the initial boot stage and throughout its entire operation. *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 8712-MOD-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM

Benefits of immutable bootstrap configurations

Immutable bootstrap configurations significantly benefit network device operation and management.

- Security and stability: Since the immutable bootstrap configuration remains unchanged after the initial setup, it ensures that the device's foundational settings are secure and stable. This reduces the risk of unauthorized changes or accidental misconfigurations.
- Consistent boot up: By maintaining a static configuration, the device can consistently boot into a known good state. This consistency is crucial for reliable network operations and troubleshooting.

Key features of immutable bootstrap configurations

These are the key features of immutable bootstrap configuration:

- Upon successful Bootz completion, the boot configuration file is stored as backup in the router. The router preserves this backup file across reloads. The boot configuration commit appears in the router's commit history as the user **local** in the output of the **show configuration commit list** command.
- The router does not allow gNMI to override the boot configurations.
- The router does not allow configuration changes for security credentials managed by gRPC Network Security Interface (gNSI) to be overwritten by gNMI or boot configuration. This means that gNSI owns the configurations for security credentials, and you must invoke gNSI to mutate this store.
- The router allows configuration changes through Command Line Interface (CLI) to take effect on the boot configuration. However if you perform a Route Processor Fail Over (RPFO), gNMI **Set** operation, or restart the Extensible Manageability Services Daemon (EMSD) process using the **process restart**

emsd command, the boot configuration is restored. The EMSD process is the process that initializes and manages the gRPC server and services on the router.

- The gNOI.Bootz service manages boot configuration store completely. If you need to modify the boot configuration you must call the **Set** RPC through the **gNOI bootconfig** service.

Limitations of immutable bootstrap configuration

These are the limitations of immutable boostrap configuration:

- When you modify any of the parameters of the boot configuration using CLI, the router applies the changes. To revert the boot configuration to the original state, perform any one of the these operations:
  - RPFO
  - gNMI **Set** operation, or
  - restart of the EMSD process using the **process restart emsd** command.
- Due to the absence of a vendor-neutral boot configuration data set, the immutable bootstrap configuration may differ depending on the vendor of the network device.
- Boot configurations cannot be altered via gNMI set operations. This restriction is vital for safeguarding critical settings like hardware-module configurations and user credentials from accidental removal.

Verify immutable bootstrap configuration on the router

Use this procedure to view and verify the immutable bootstrap configurations.

Before you begin

Ensure the router is booted up and provisioned with the Bootz process.

Procedure

Step 1 Enter the **show configuration commit list** command to view the list of configurations commited.

Example:

```
Router# sh configuration commit list 3
SNo. Label/ID              User      Line               Client      Time Stamp
~~~~ ~~~~~~~~              ~~~~      ~~~~               ~~~~~~      ~~~~~~~~~~
1    1000000011            local     ems                YANG Frame  Thu Nov 14 12:26:26 2024
2    1000000010            cisco     con0_RP0_CPU0      CLI         Thu Nov 14 12:23:14 2024
3    1000000009            local     ems                YANG Frame  Thu Nov 14 05:55:50 2024
```

The router commits the bootstrap configuration as User **local**.

From the output of **show configuration commit list**, note the specific **Label/ID** listed for the User **local**.

Step 2 Enter the **show configuration commit changes** command with the **Label/ID** noted in the previous step, to view the list of bootstrap configurations.

Example:

```
Router# sh configuration commit changes 1000000011
!! Building configuration...
!! IOS XR Configuration 24.4.1
hostname cisco
username cisco
 group root-lr
 group cisco-support
 secret 10
$6$8J3HfJlo8Ah0f...$gG5juuKsunhYwcH1rRRdq2UHPk/VUxVGjVHRLAhrUFlyOoiBRjQFVdDhgYlfuOJDRmAfLv1H4AaOZX8xHrBp20
!
grpc
 no-tls
!
interface MgmtEth0/RP0/CPU0/0
 ipv4 address dhcp
!
router static
 address-family ipv4 unicast
 0.0.0.0/0 MgmtEth0/RP0/CPU0/0 192.168.122.1
 !
!
ssh server v2
end
```

Bootz dynamic configuration

Bootz dynamic configuration is a set of boot configuration fields that

- applies initial device settings that can be changed after bootstrap
- supports automation and user-driven updates via standard methods like CLI or gNMI, and
- differs from immutable configurations that remain locked for device security and operational stability.

Table 21: Feature History Table

Feature Name	Release Information	Feature Description
Bootz dynamic configuration	Release 26.3.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q100, Q200, P100], 8700 [ASIC: P100, K100], 8010 [ASIC: A100]); Centralized Systems (8600 [ASIC: Q200], 8400 [ASIC: K100]); Modular Systems (8800 [LC ASIC: Q100, Q200, P100, K100]) You can now set initial device configuration at first boot that remains flexible for future changes. This feature allows you to modify or overwrite dynamic boot-time settings through standard CLI or gNMI Set RPC calls without requiring a factory reset.

Dynamic versus immutable Bootz configurations

From IOS XR 26.3.1, Bootz defines two categories of configuration at device bootstrap, each with specific characteristics:

- **Immutable configuration:**

  Boot-time vendor and OpenConfig settings that are preserved across device reloads and cannot be changed by subsequent configuration actions.

- **Dynamic configuration:**

  Boot-time settings that are mutable after bootstrap, supporting post-boot automation and user-driven updates.

Bootz processes configurations in two phases during device startup: it first applies and preserves immutable Cisco IOS XR native and OpenConfig data, then applies the dynamic settings as regular gNMI operations. Bootz dynamic configurations are applied using fields within the BootConfig message defined in Bootz.proto.

The following table compares essential attributes of dynamic and immutable configuration:

Table 22: Comparison of dynamic and immutable configuration attributes

Attribute	Dynamic configuration	Immutable configuration
Applied at bootstrap	Yes	Yes
Mutable after device boot	Yes	No
Preserved across device reload or RP failover	No	Yes
Configuration update requires factory reset	No	Yes

Note Use dynamic configuration for settings that need to change after initial boot. Use immutable config for parameters that must remain locked for device security and operational stability.

Initial device deployment with dynamic and immutable configuration

When a new Cisco router boots, BootZ delivers both `vendor_config` and `dynamic_vendor_config` fields.

- The `vendor_config` is immutable and sets security and identity.
- The `dynamic_vendor_config` is mutable and provides BGP neighbors and interface addresses needed for deployment. After boot, automation updates BGP neighbors with gNMI `Set`, demonstrating that dynamic configuration can change without reapplying the bootstrap.

Guidelines for Bootz dynamic configuration

Configuration category selection

Reserve dynamic configuration for parameters that require future modification or automation-driven updates. Use immutable configuration fields for device settings that must not change after bootstrap, such as security policies or device identity.

Dynamic configuration consumption and persistence

Provide dynamic configuration fields only during the initial Bootz bootstrap event in the BootstrapData message.

- Do not specify dynamic config with post-boot gNOI SetBootConfig calls, as the router rejects them.
- Dynamic configuration is applied only if the bootconfig.txt file is present, which signals the device is in first-time bootstrap state.
- After initial application, dynamic config options are not consumed again, even during device restarts or route processor failovers.

Error handling and rollback

Rely on the EMSD process to enforce atomicity of bootstrap configuration. If dynamic configuration application fails, EMSD rolls back both dynamic and immutable configuration settings to protect device integrity.

- Always validate configuration data before bootstrap to lower the risk of error-triggered rollback.
- Configuration failures in dynamic or immutable sections trigger rollback and error reporting to external systems by Bootz status messages and EMSD logs.
- Operational teams should monitor EMSD artifact status files for each bootstrapping component to verify results and troubleshoot failures.

Security considerations for dynamic configuration

Apply operational security policies and validations to all dynamic configuration fields as you would with immutable configuration.

- Validate and approve all configuration content, both dynamic and immutable, before assembling BootConfig messages for device provisioning.
- Design automation and operational workflows to detect and alert on unexpected configuration artifacts or status changes during or after bootstrap.
- Bootz dynamic configuration does not introduce new external attack surfaces, but misconfiguration or using unvalidated dynamic settings may create operational risks.

How Bootz dynamic configuration works

Bootz dynamic configuration enables Cisco routers to process both dynamic and immutable configuration during device bootstrap. It facilitates rapid onboarding with flexible initial settings and protecting critical persistent parameters.

Summary

The key components involved in the process are:

- Bootz client: Retrieves and stages both immutable and dynamic bootstrap artifacts on the device.
- EMSd: Applies configuration artifacts, reports status, manages rollback on error, and updates status records.
- ZTP (Zero Touch Provisioning): Initiates automated bootstrap by communicating with infrastructure at initial device boot.

Bootz dynamic configuration ensures configuration artifacts are applied in sequence with robust error handling, guaranteeing that both immutable and dynamic configurations are properly delivered and that operational status remains traceable.

Workflow

These stages describe how Bootz dynamic and immutable configuration are processed during device bootstrap:

1. On device startup, ZTP and the Bootz client initiate bootstrap by acquiring initial configuration and credentials from the Bootz server.
   - ZTP triggers the Bootz flow using DHCP, obtaining bootstrap data (BSD) from the network.
   - The Bootz client fetches the BSD and validates all artifacts for authenticity.

   All bootstrap artifacts are validated before configuration is applied.

   The device is prepared to stage configuration files.
2. The Bootz client extracts, validates, and stages artifacts on local storage.
   - Artifacts include vendor_config, oc_config, dynamic_vendor_config, dynamic_oc_config, credentials, and policies.
   - Artifacts are classified as immutable (unchangeable) or dynamic (modifiable after the initial boot).

   The artifacts become available for EMSd to apply to the device configuration system.

   All required files are staged for EMSd processing.
3. EMSd loads artifact files and applies configurations in two steps: immutable artifacts first, followed by dynamic artifacts.
   - Immutable configuration (vendor_config, oc_config) is applied first using a gNMI Set request with priority 1.
   - Upon successful immutable application, dynamic configuration (dynamic_vendor_config, dynamic_oc_config) is applied using a separate gNMI Set request with priority 100.

   EMSd maintains the proper application order for consistency and security.

   Immutable settings are locked until device reset; dynamic settings become part of the running configuration.
4. EMSd validates each configuration application, manages rollbacks, and records status for each artifact type.
   - EMSd detects errors when configuring artifacts and initiates rollback for both immutable and dynamic configurations if required.

- Status for each artifact is written, such as for dynamic configuration, in `/misc/config/ems/gnsi` and system logs.

When...	And...	Then...	And...
A configuration artifact fails to apply	The device is in bootstrap phase	EMSd rolls back all applied artifacts (immutable and dynamic)	An error status is sent to the Bootz client and logged
All artifacts apply successfully	The device reaches operational state	EMSd marks all artifacts as OK in status files	External management services are enabled

Status files are referenced for troubleshooting and provide per-artifact reporting.

The device completes bootstrap with validated immutable and dynamic configuration in effect.

5. After bootstrap and across device restarts, only immutable configuration is enforced - dynamic configuration is not reapplied. So user updates persist after the first boot.
   - Dynamic configuration from bootconfig.txt is used only during first boot. For subsequent restarts, immutable configuration from bootconfig_merged.txt is applied.
   - Operators can change dynamic elements using gNMI Set or CLI.

   This protects operator changes from being overwritten after bootstrap and enforces critical configuration until reset.

   The device supports further dynamic configuration after deployment but ensures immutable parameters are preserved until factory reset.

Result

When the process completes, the router is securely bootstrapped with immutable and dynamic configuration - enabling reliable onboarding and ongoing operational flexibility. All status is recorded and visible for troubleshooting and compliance tracking.

Verify Bootz dynamic configuration

Verify Bootz dynamic configuration is successfully applied during device bootstrap and remains mutable.

This task helps network operators verify successful BootZ bootstrapping and diagnose configuration or artifact failures using device logs and status files.

Before you begin

- Ensure the router is booting from a factory or unconfigured state, or is being bootstrapped using Bootz.
- Confirm that the Bootz server, DHCP, and required ZTP infrastructure are reachable and properly configured.
- Make sure you can access the device shell or management interface to inspect logs and configuration files.

Follow these steps to verify and troubleshoot Bootz dynamic configuration.

Procedure

Step 1 Review ZTP logs to check bootstrap status.

Example:

```
Router# tail -f /var/log/ztp.log
```

Check for messages indicating successful completion or errors in bootstrapping, such as **Bootz OK**, **Completed**, or **Error**.

Table 23: ZTP log review outcomes

If...	Then...
Completion messages and no errors appear	Go to the next step to verify artifact status.
Error or rollback messages appear	Review logs in detail and check EMSd status for artifact failures.
Log file is missing or unreadable	Verify bootstrapping was properly initiated and confirm device access.

ZTP log review provides initial confirmation of Bootz bootstrapping success or failure.

Step 2 Compare running configuration output with the initial dynamic configuration to confirm correct application.

Example:

```
Router# show running-config
```

Table 24: Dynamic configuration verification outcomes

If...	Then...
Dynamic configuration entries are present	Dynamic configuration applied successfully and continue to next step.
Dynamic configuration entries are missing	Troubleshoot artifact status and check application logs.
Unexpected configuration present	Review original bootstrap artifacts for discrepancies.

Running configuration output indicates if dynamic configuration was applied at bootstrap.

Step 3 Change a value set by dynamic configuration using the CLI or gNMI Set and verify the change appears in running config.

Example:

```
gnmi-cli set dynamic-config-path value new-value
```

Immutable config cannot be changed; dynamic config should be updatable.

Table 25: Mutability outcomes

If...	Then...
Change to dynamic configuration is accepted and reflected	Dynamic configuration mutability is verified.
Change is rejected	Confirm the configuration path is dynamic and review error messages.
Immutable configuration path is changed	Change should be rejected by the device.

Mutability test confirms that dynamic configuration can be changed, while immutable configuration cannot.

Step 4 Run the **show logging** command to look for rollback, commit failures and bootconfig application failure messages.

Example:

```
Router# show logging
```

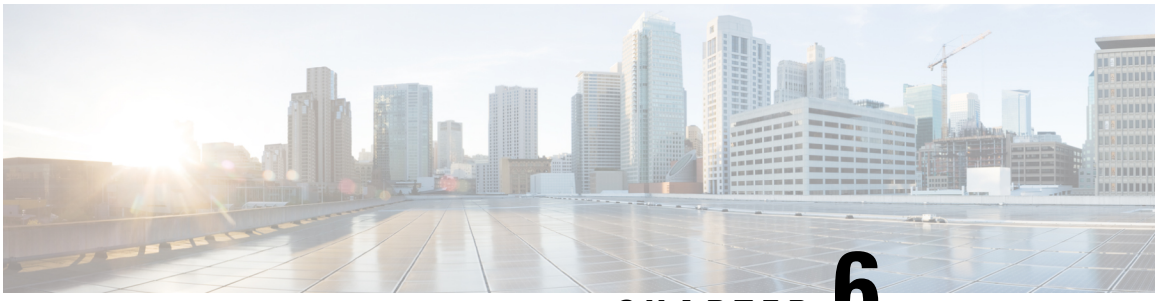
Error logs help isolate failures in configuration application, supporting troubleshooting efforts.

If verification succeeds, the router operates with Bootz status as OK, immutable and dynamic configuration artifacts present, dynamic configuration is mutable, and errors during bootstrap result in rollback and clear error reporting.

What to do next

After you finish verification and troubleshooting:

- Monitor device state for further changes and ensure that operational workflows can update dynamic config as needed.
- Document troubleshooting outcomes if errors occurred and escalate to Cisco support if bootstrap or persistent errors remain.



CHAPTER 6

Deploy Router Using Secure ZTP

With Secure Zero Touch Provisioning (ZTP), you can securely and seamlessly provision thousands of network devices accurately within minutes and without any manual intervention.

Table 26: Feature History Table

Feature	Release Information	Feature Description
Secure Zero Touch Provisioning with Removable Storage Device	Release 25.4.1	Introduced in this release on: Fixed Systems (8010 [ASIC: A100])(select variants only*) *This feature is supported on: • 8011-32Y8L2H2FH • 8011-12G12X4Y-A/D
Secure Zero Touch Provisioning with Removable Storage Device	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM

Feature	Release Information	Feature Description
Secure Zero Touch Provisioning	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM
Secure Zero Touch Provisioning with Removable Storage Device	Release 7.3.2	This feature allows you to securely sign onboarding data in a removable storage device so that you can use the device for secure ZTP operations. This support gives you the plug-and-play flexibility for ZTP without any additional infrastructure requirements.
Secure Zero Touch Provisioning	Release 7.3.1	This feature allows devices in the network to establish a secure connection with the ZTP server and authenticate information using a three-step validation process involving validation of the network device, the ZTP server, and onboarding information. This eliminates security risks or malicious actions during remote provisioning. The **ztp secure-mode enable** command is introduced.

In a secured network such as datacenter, the zero-touch provisioning mechanism helps you provision hundreds of remote devices without your intervention. But, the access devices are typically in an insecure network. There is a high risk of malicious actions on the device, such as adding an unauthorized or infected device. Security is a critical aspect while remotely provisioning the network devices.

Secure ZTP combines seamless automation with security. Network devices can securely establish a connection with the ZTP server and authenticate the onboarding information that it receives. The process eliminates any security risks or malicious actions during the provisioning of remote devices.

- ZTP helps you remotely provision a router securely anywhere in the network. Thus, eliminate the risk of malicious attacks or unauthorized ownership claims.
- Secure ZTP authenticates not only the onboarding network device but also validates the server authenticity and provisioning information that it is receiving from the ZTP server.

Cisco IOS XR software implements the secure zero touch provisioning capabilities as described in RFC 8572. Secure ZTP uses a three-step validation process to onboard the remote devices securely:

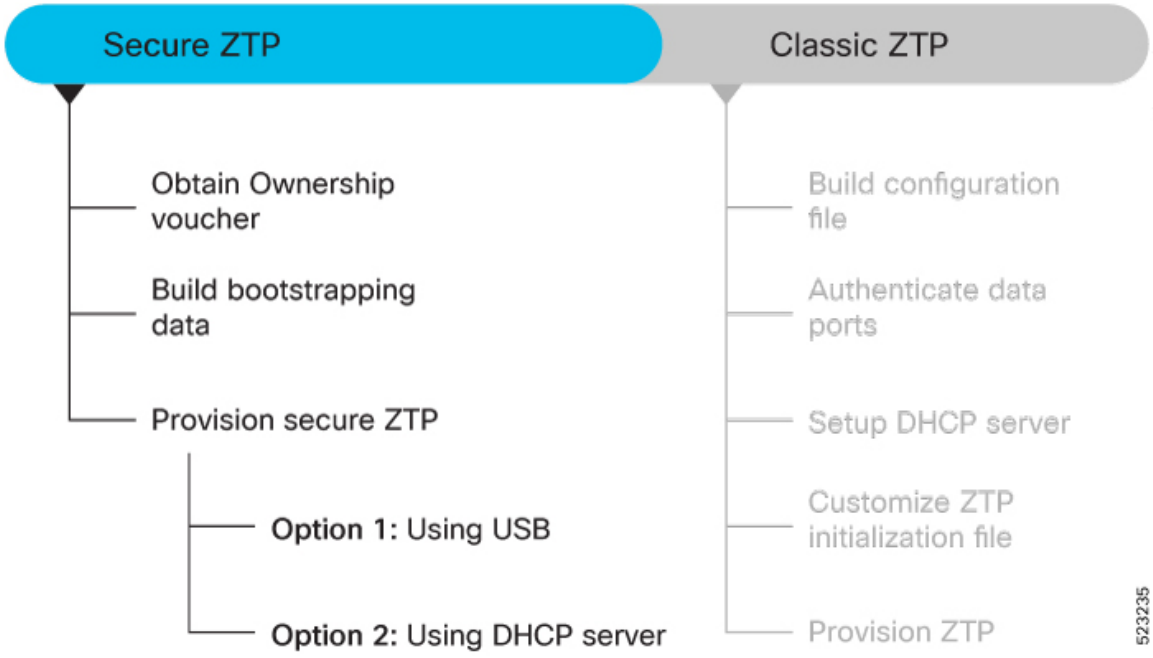
1. Router Validation: The ZTP server authenticates the router before providing bootstrapping data using the Trust Anchor Certificate (also called SUDI certificate).

Note The gRPC server advertises TLS 1.2 as the maximum TLS version when using a SUDI-based SSL profile.

2. Server Validation: The router device in turn validates the ZTP server to make sure that the onboarding happens to the correct network. Upon completion, the ZTP server sends the bootstrapping data (for example, a YANG data model) or artifact to the router.
3. Artifact Validation: The configuration validates the bootstrapping data or artifact received from the ZTP server.

Follow the workflow to understand the tasks involved in provisioning the router using secure ZTP.

Figure 16: Secure ZTP Workflow



This section contains the following topics:

- Obtain Ownership Voucher, on page 102
- Build Bootstrapping Data, on page 103
- Secure ZTP Options, on page 106

Obtain Ownership Voucher

The ownership voucher is used to identify the owner of the device by verifying the owner certificate stored in the device.

Table 27: Feature History Table

Feature	Release Information	Feature Description
Ownership voucher request enhancement	Release 25.3.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q200, P100], 8700 [ASIC: P100, K100]); Modular Systems (8800 [LC ASIC: Q100, Q200, P100]) From this release onwards, when requesting an ownership voucher for fixed systems, you must use the chassis serial numbers in the Ownership Voucher request. Additionally, for both modular and fixed chassis systems, the serial number included in the ownership voucher request must match the SUDI subject name or the certificate on the device. This ensures accurate device identification and owner verification.

How to obtain Ownership Voucher

These steps help you obtain the ownership voucher from Cisco:

1. Contact Cisco Support.
2. Provide these information in your request to Cisco.
   - **Pinned Domain certificate (PDC):** PDC is an X.509 v3 certificate structure that uses Distinguished Encoding Rules (DER). The router uses this certificate to trust a public key infrastructure for verifying a domain certificate supplied to the router separately in the bootstrapping data. This certificate could be an end-entity certificate, including a self signed entity.
   - Purchase order details with the serial numbers of the routers.

   Sample Request:

```
{
   "expires-on": "2016-10-21T19:31:42Z",
   "assertion": "verified",
   "serial-number": "JADA123456789",
   "idevid-issuer": "base64encodedvalue==",
   "pinned-domain-cert": "base64endvalue==",
   "last-renewal-date": "2017-10-07T19:31:42Z"
}
```

3. Cisco generates the ownership voucher in .vcj format (Example: DCA213140YX.vcj) and sends the voucher in response to your request.

Note

- On fixed systems, use chassis serial number in the Ownership Voucher request.
- On modular systems, use Route Processor serial number in the Ownership Voucher request.

Build Bootstrapping Data

The following describe the components of secure ZTP:

- **Onboarding Device (Router):** The router is a Cisco device that you want to provision and connect to your network. Secure ZTP is supported only on platforms that have Hardware TAM support. Routers with HW TAM have the SUDI embedded in TAM.
- **DHCP Server:** The secure ZTP process relies on the DHCP server to provide the URL to access the bootstrapping information.
- **ZTP Server:** A ZTP server is any server used as a source of secure ZTP bootstrapping data and can be a RESTCONF or HTTPs server.

Note ZTP only supports single name-server. When the DHCP server has more than one server address configured, ZTP fails to apply the server configuration.

The ZTP server contains the following artifacts:

- Cisco IOS XR software images: You can download Cisco images, SMU, and patches using the Cisco Support & Downloads page.
- ZTP scripts: Contains the following libraries and you can build a script to initiate the ZTP process.
  - Python library: Includes IOS XR CLI (show commands and configuration commands), YANG-XML (ncclient, native Netconf client), and YANG-JSON (gnmic or gNMI client)).

Note Update your ZTP scripts from python2 to python3, which is supported from Cisco IOS XR Release 24.1.1. You can also use python2-based ZTP scripts until Cisco IOS XR Release 25.2.1.

  - BASH library: Includes IOS XR CLI show commands, configuration commands
  - Bootstrapping Data

- **Bootstrapping Data:** It is the collection of data that the router obtains from the ZTP server during the secure ZTP process. You must create and upload the bootstrapping data in the ZTP server. For more information, refer RFC 8572.
  - The bootstrapping data mainly has three artifacts:

- **Conveyed Information**: Conveyed Information contains the required bootstrapping data for the device. It contains either the redirect information or onboarding information to provision the device.

  For example:

```
module: ietf-sztp-conveyed-info

          yang-data conveyed-information:
            +-- (information-type)
               +--:(redirect-information)
               |  +-- redirect-information
               |     +-- bootstrap-server* [address]
               |        +-- address         inet:host
               |        +-- port?           inet:port-number
               |        +-- trust-anchor?   cms
               +--:(onboarding-information)
                  +-- onboarding-information
                     +-- boot-image
                     |  +-- os-name?              string
                     |  +-- os-version?           string
                     |  +-- download-uri*         inet:uri
                     |  +-- image-verification* [hash-algorithm]
                     |     +-- hash-algorithm    identityref
                     |     +-- hash-value        yang:hex-string
                     +-- configuration-handling?      enumeration
                     +-- pre-configuration-script?    script
                     +-- configuration?               binary
                     +-- post-configuration-script?   script
```

  - **Redirect Information**: Redirect information is used to redirect a device to another bootstrap server. The redirect information contains a list of bootstrap servers along with a hostname, an optional port, and an optional trust anchor certificate that the device uses to authenticate the bootstrap server.

    For Example:

```
{
     "ietf-sztp-conveyed-info:redirect-information" : {
       "bootstrap-server" : [
         {
           "address" : "sztp1.example.com",
           "port" : 8443,
            "trust-anchor" : "base64encodedvalue=="
         },
         {
           "address" : "sztp2.example.com",
           "port" : 8443,
           "trust-anchor" : "base64encodedvalue=="
         },
         {
           "address" : "sztp3.example.com",
           "port" : 8443,
           "trust-anchor" : "base64encodedvalue=="
         }
       ]
     }
   }
```

  - **Onboarding Information:** Onboarding information provides data necessary for a device to bootstrap itself and establish secure connections with other systems. It specifies details

about the boot image, an initial configuration the device must commit, and scripts that the device must execute.

For Example:

```
{
     "ietf-sztp-conveyed-info:onboarding-information" : {
       "boot-image" : {
         "os-name" : "VendorOS",
         "os-version" : "17.2R1.6",
         "download-uri" : [ "https://example.com/path/to/image/file" ],
         "image-verification" : [
           {
             "hash-algorithm" : "ietf-sztp-conveyed-info:sha-256",
             "hash-value" : "ba:ec:cf:a5:67:82:b4:10:77:c6:67:a6:22:ab:\
   7d:50:04:a7:8b:8f:0e:db:02:8b:f4:75:55:fb:c1:13:b2:33"
           }
         ]
       },
       "configuration-handling" : "merge",
       "pre-configuration-script" : "base64encodedvalue==",
       "configuration" : "base64encodedvalue==",
       "post-configuration-script" : "base64encodedvalue=="
     }
   }
```

- **Owner Certificate**: The owner certificate is installed on the router with the public key of your organization. The router uses the owner certificate to verify the signature in the conveyed information artifact using the public key that is available in the owner certificate.
- **Ownership Voucher**: Ownership Voucher is used to identify the owner of the device by verifying the owner certificate that is stored in the device. Cisco supplies Ownership Voucher in response to your request. You must submit the Pinned Domain Certificate and device serial numbers with the request. Cisco generates and provides the Ownership Voucher to you.

- **Report Progress:** When the device obtains the onboarding information from a ZTP server, the router reports the bootstrapping progress to the ZTP server using the API calls.

  See RFC 8572 for the detailed report-progress messages that can be sent to the ZTP server.

  The following is the structure of the `report-progress` sent the progress message to a ZTP server.

```
+---x report-progress {onboarding-server}?
          +---w input
             +---w progress-type         enumeration
             +---w message?              string
             +---w ssh-host-keys
             |  +---w ssh-host-key* []
             |     +---w algorithm    string
             |     +---w key-data     binary
             +---w trust-anchor-certs
                +---w trust-anchor-cert*   cms
```

  The following example illustrates a device using the Yang module to post a progress report to a ZTP server with a `bootstrap complete` message:

```
{
        'progress-type': 'bootstrap-complete',
        'message': 'example message',
        'trust-anchor-certs': [{
```

```
                'trust-anchor-cert': 'base64encodedvalue=='
        'ssh-host-keys': [{
                'key-data': 'base64encodedvalue==',
                'algorithm': 'ssh-rsa'
        }, {
                'key-data': 'base64encodedvalue==',
                'algorithm': 'rsa-sha2-256'
        }]
}

RESPONSE from the ZTP server

   HTTP/1.1 204 No Content
   Date: Sat, 31 Oct 2015 17:02:40 GMT
   Server: example-server
```

Secure ZTP Options

Provision secure ZTP using a USB drive

Enable secure, automated onboardin of a Cisco IOS XR device using a removable USB drive, ensuring all bootstrapping data is authenticated and provisioned.

A USB drive used for device provisioning is considered an untrusted source and must contain sined onboarding data. For secure ZTP, the USB drive must included the following artifacts, all signed:

- Conveyed Information
- Owner Certificate
- Ownership Voucher

These artifacts enable the device to verify authenticity, automate configuration, and maintain security standards.

Follow these steps to provision secure ZTP:

Before you begin

- Enable secure ZTP on the router using the **ztp secure-mode enable** command and reload the router.
- Obtain an Ownership Voucher from Cisco Support. Provide a *Pinned Domain certificate (PDC)*[3] and router serial numbers to Cisco Support.
- On fixed systems, use the chassis serial number; on modular systems, use the Router Processor serial number.

  Example Ownership Voucher request:

```
{
                        "expires-on": "2016-10-21T19:31:42Z",
                        "assertion": "verified",
                        "serial-number": "JADA123456789",
                        "idevid-issuer": "base64encodedvalue==",
                        "pinned-domain-cert": "base64endvalue==",
```

[3] PDC is an X.509 v3 certificate structure that uses Distinguished Encoding Rules (DER). This certificate is used by the router to trust a public key infrastructure in order to verify a domain certificate supplied to the router separately in the bootstrapping data. This certificate could be an end-entity certificate, including a self signed entity.

```
"last-renewal-date": "2017-10-07T19:31:42Z"
}
```

Procedure

Step 1 Copy the following signed onboarding artifacts to the `EN9` direct at the root of the USB drive:

- `conveyed-information.cms`: Contains onboarding information (IOS XR images, configuration scripts).
- Download images and patches from the Cisco Support & Downloads portal.
- Build ZTP scripts using Python or Bash for IOS XR CLI commands and configuration.

  Note
  Update your ZTP scripts from python2 to python3, which is supported from Cisco IOS XR Release 24.1.1. You can also use python2-based ZTP scripts until Cisco IOS XR Release 25.2.1 .

- `owner-certificate.cms`: The signed owner certificate.
- `ownership-voucher.vcj`: The ownership voucher (obtained from Cisco Support).

Step 2 Plug the USB drive into the router.

Step 3 Power ON the router.

Step 4 The device automatically verifies the secure ZTP mode is enabled, and scans the USB for the `EN9` directory and required artifacts.

Here is the high-level workflow of the Secure ZTP process using a removable storage device:

a. When you boot the device with an IOS-XR image, the secure ZTP process verifies if the secure ZTP mode (`secure-ztp mode`) is enabled. If not enabled, the device boots normally.

b. The device verifies if the USB is enabled in the `ztp.ini` file. By default, the USB is enabled and assigned the highest priority in the fetcher priority in the `ztp.ini` file.

Fetcher priority defines how secure ZTP can get the provisioning details. By default, each port has a fetcher priority defined in the `ztp.ini` file. The fetcher priority range is from 0 to 9. The lower the number higher is the priority. The value 0 has the highest priority and 9 has the lowest priority.

The following example shows the sample of the `ztp.ini` file:

```
[Startup]
start: True
retry_forever: True
[Fetcher Priority]
USB: 0
Mgmt4:  1
Mgmt6:  2
DPort4: 3
DPort6: 4
```

c. Secure ZTP checks for a USB drive on the router. If the USB drive is not available, the secure ZTP process moves to the next fetcher as defined in the fetcher priority of the `ztp.ini` file.

d. If a USB drive is available, the router scans for the `EN9` directory in the root of the USB drive.

If the `EN9` directory is not available, the secure ZTP process moves to the next fetcher as defined in the fetcher priority of the `ztp.ini` file.

Step 5 The router validates all onboarding artifacts received from the USB drive:

- Extracts and verifies the `pinned-domain-cert` from the ownership voucher.
- Authenticates the owner certificate through X.509 path verification on the trusted certificate.
- Confirms that the conveyed information is signed by the validated owner certificate.
- Checks that the owner certificate verifies the conveyed information.

Step 6 The router provisions itself:

- Processes boot image information.
- Executes preconfiguration and post-configuration scripts.

  Commits the initial configuration.

The Cisco IOS XR device is securely provisioned and operational. If any artifact validation fails or the required directory is missing, the device moves to the next available provisioning method as defined in its fetcher priority settings of the `ztp.ini` file.

What to do next

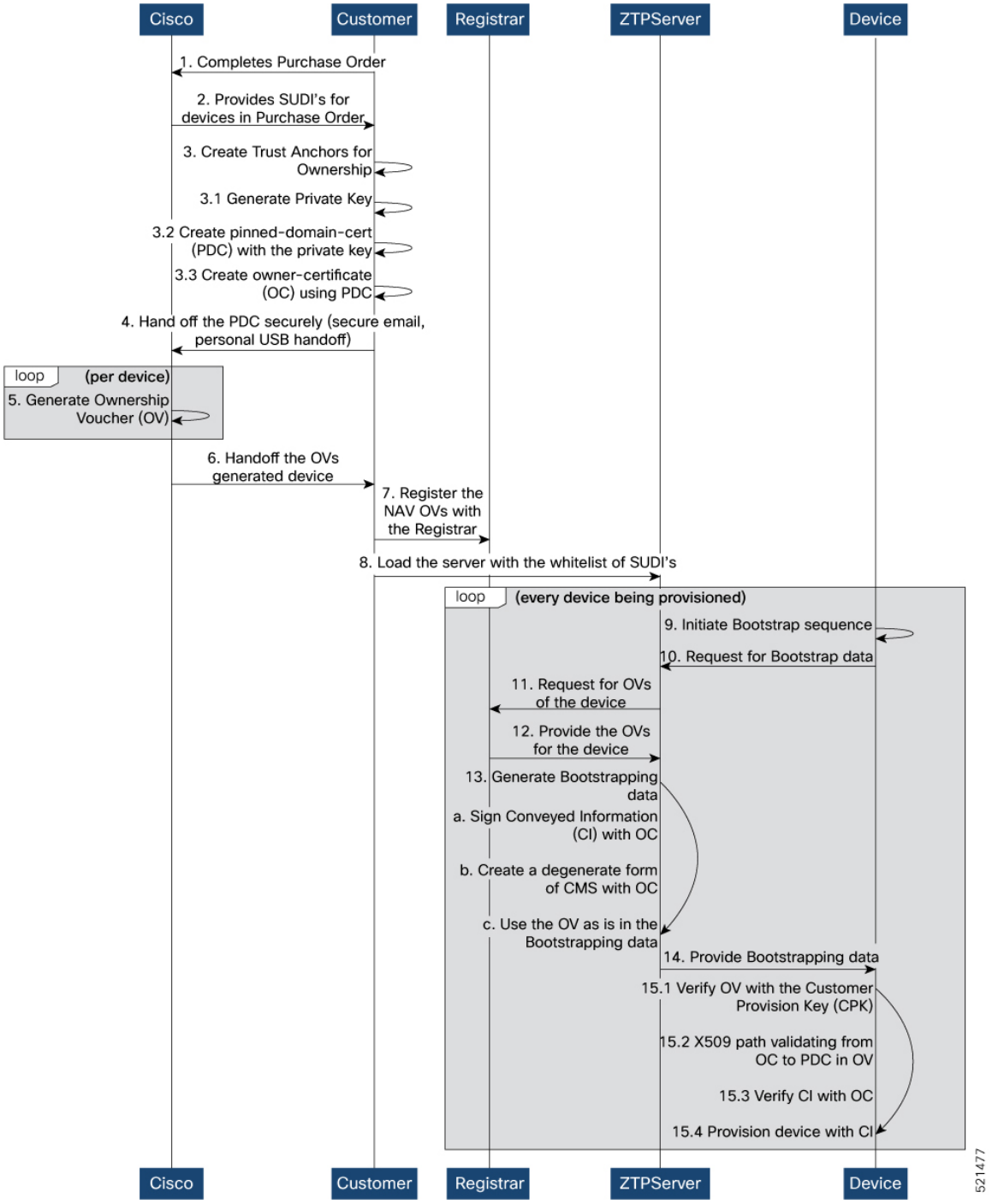
- Confirm device operation by reviewing logs and verifying configurations.
- Optionally, reuse the USB drive for additional device setups as needed.

Provision Secure ZTP Using DHCP Server

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.

The following figure illustrates the end-to-end sequence of the Secure ZTP process:

Figure 17: End-to-end sequence of the Secure ZTP process



Before you begin

The network administrator performs the following tasks as part of the initial setup for secure ZTP:

- Ensure to enable secure ZTP on the router using the **ztp secure-mode enable** command and then reload the router.
- Contact Cisco Support to obtain a voucher. Provide the following details to request for ownership voucher certificate:
  - Pinned Domain certificate (PDC): PDC is an X.509 v3 certificate structure that uses Distinguished Encoding Rules (DER). This certificate is used by the router to trust a public key infrastructure in order to verify a domain certificate supplied to the router separately in the bootstrapping data. This certificate could be an end-entity certificate, including a self signed entity.
  - Order details with the Serial numbers of the routers

  For example,

```
{
        "expires-on": "2016-10-21T19:31:42Z",
        "assertion": "verified",
        "serial-number": "JADA123456789",
        "idevid-issuer": "base64encodedvalue==",
        "pinned-domain-cert": "base64endvalue==",
        "last-renewal-date": "2017-10-07T19:31:42Z"
    }
```

Procedure

Step 1 Upload the following bootstrapping data to the ZTP server. Steps to upload may vary depending on the server that you're using, refer to the documentation provided by your vendor.

- Cisco IOS XR software images: You can download Cisco images, SMU, and patches using the Cisco Support & Downloads page.
- ZTP scripts that include IOS XR configurations, pre, and post configuration scripts. Build a script to initiate the ZTP process. See Build Configuration File, on page 118.
  - Python library: Includes IOS XR CLI (show commands and configuration commands) and YANG-XML (`ncclient`, `native Netconf client`).

    Note
    Update your ZTP scripts from python2 to python3, which is supported from Cisco IOS XR Release 24.1.1. You can also use python2-based ZTP scripts until Cisco IOS XR Release 25.2.1.

  - BASH library: Includes IOS XR CLI show commands, configuration commands
- Serial numbers of the routers you plan to onboard using ZTP
- Owner certificates
- Pinned Domain Certificate (PDC)
- Ownership vouchers

Step 2 Set up the DHCP server to provide the redirect URL to the router:

Before triggering the secure ZTP process, configure the DHCP server to provide the location of the IOS-XR image to the router. For information on how to configure the DHCP server, see your DHCP server documentation.

Configure the following parameters in the DHCP server:

- `option-code`: The DHCP SZTP redirect Option has the following parameters:
  - OPTION_V4_SZTP_REDIRECT (143): Use this DHCP v4 code for IPV4.
  - OPTION_V6_SZTP_REDIRECT (136): Use this DHCP v4 code for IPV6.

  For example, `option dhcp6.bootstrap-servers code 136 = text;`
- `option-length`: The option length in octets
- `bootstrap-servers`: A list of servers for the onboarding device to contact the servers for the bootstrapping data.
- `bootfile-url` : The URI of the SZTP bootstrap server should use the HTTPS URI scheme and it should be in the following format:

  `"https://<ip-address-or-hostname>[:<port>]".`

Step 3 Power on the router.

Here is the high-level workflow of the Secure ZTP process using a removable storage device:

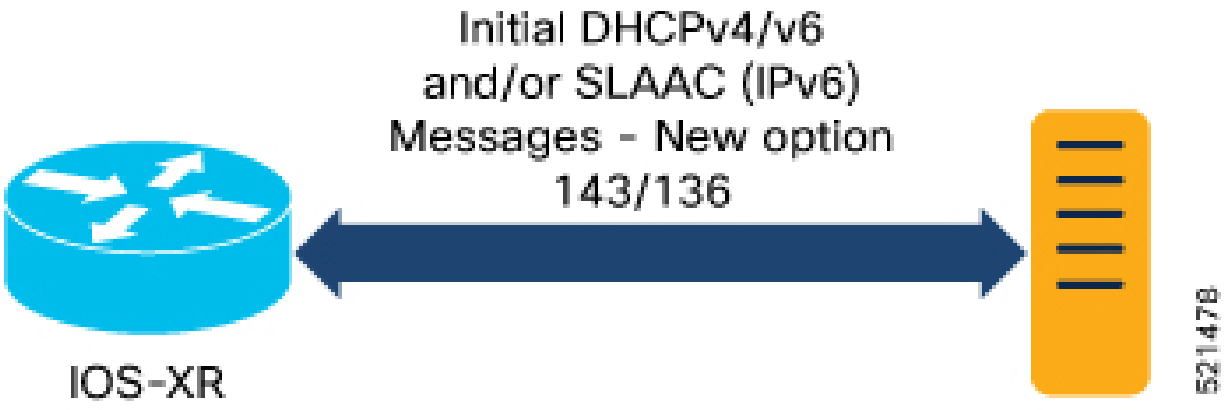
a. When you boot the device with an IOS-XR image, the secure ZTP process verifies if the secure ZTP mode (`secure-ztp mode`) is enabled. If not enabled, the device boots normally.

Note
When `secure-ztp mode` is enabled, the ZTP process accepts only the `secure-redirect-URL` and ignores the presence of boot file name option from the DHCP response.

b. **DHCP discovery**:

1. The router initiates a DHCP request to the DHCP server.
2. The DHCP server responds with a DHCPv4 143 address option (for IPv4 addressing) or a DHCPv6 136 option (for IPv6 addressing). In addition, URLs to access bootstrap servers for further configuration is also listed.

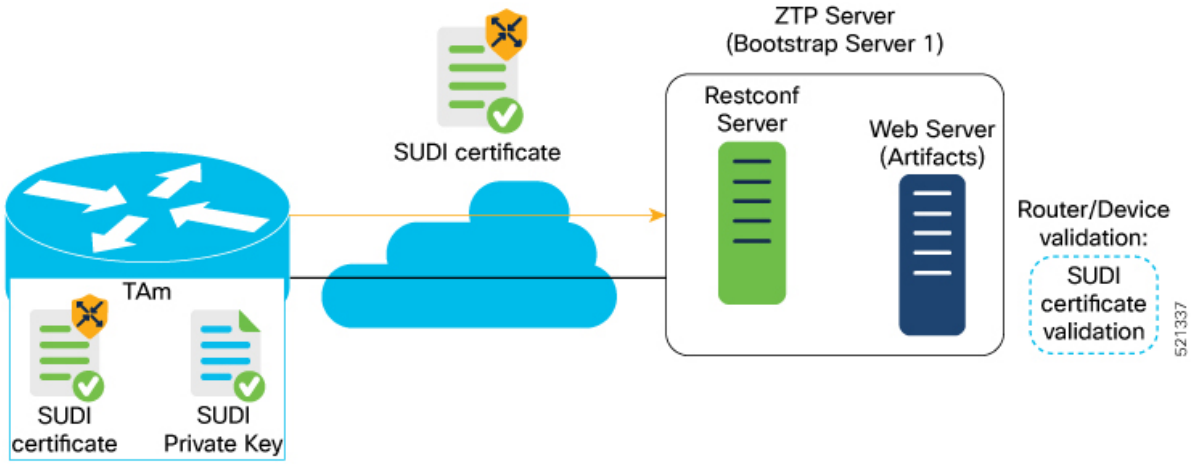
Figure 18: DHCP discovery



c. **Router validation**:

1. After receiving the URL from the DHCP server, the router sends an HTTPs request to the RESTCONF or HTTPs server using the specified URL. Along with the HTTPs request, the device sends the client certificate that is provided by the manufacturer (also called SUDI certificate). This certificate identifies and authenticates itself to the ZTP server.

Figure 19: Router Validation for Secure ZTP Provisioning

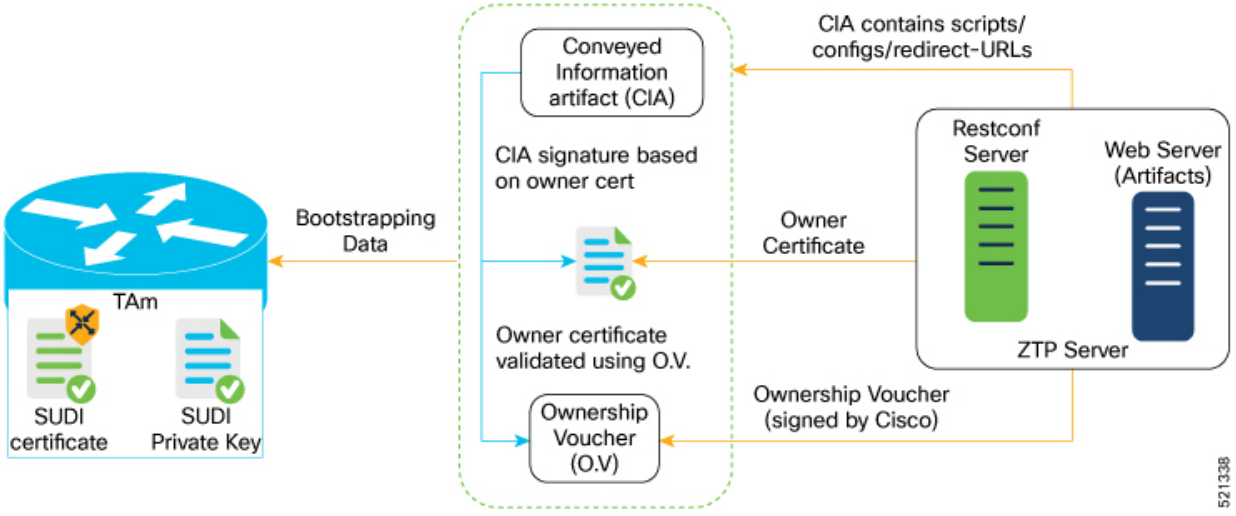


2. The RESTCONF or HTTPs server verifies the received SUDI certificate with the public certificate that it contains. Cisco issues the public certificate to ensure that the onboarding device is an authorized Cisco device.
3. After the onboarding device is authenticated, the web server sends the required artifacts along with the secure ZTP yang model to the onboarding device.

d. Server validation :

The router receives the yang model that contains Owner Certificate, Ownership Voucher, and Conveyed Information artifact. The router verifies the ownership voucher by validating its signature to one of its preconfigured trusts anchors and downloads the image. When the router obtains the onboarding information, it reports the bootstrapping progress to the ZTP server. See RFC 8572 for the progress information.

Figure 20: Server Validation for Secure ZTP Provisioning



e. Artifact Validation:

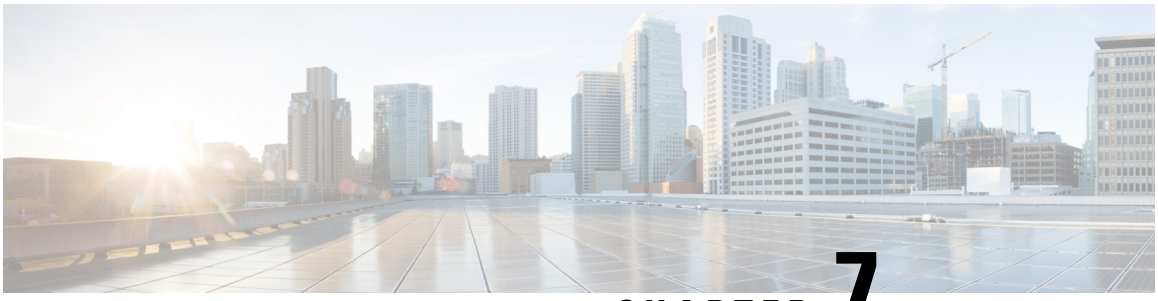
The router validates the artifact received from the ZTP server.

1. The device extracts the `pinned-domain-cert` node, an X.509 certificate from the ownership voucher to verify the owner certificate.
2. The device authenticates the owner certificate by performing the X.509 certificate path verification process on the trusted certificate.
3. Finally, the device verifies whether the conveyed information artifact is signed by the validated owner certificate.

f. Provision the device:

1. The device first processes the boot image information.
2. Executes the pre-configuration script and then commits the initial configuration
3. Execute the post configuration script.

g. After the onboarding process is completed, the network device is operational.



CHAPTER 7

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. Currently, ZTP only supports single name-server. When the DHCP server has more than one server address configured, ZTP fails to apply the server configuration.

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

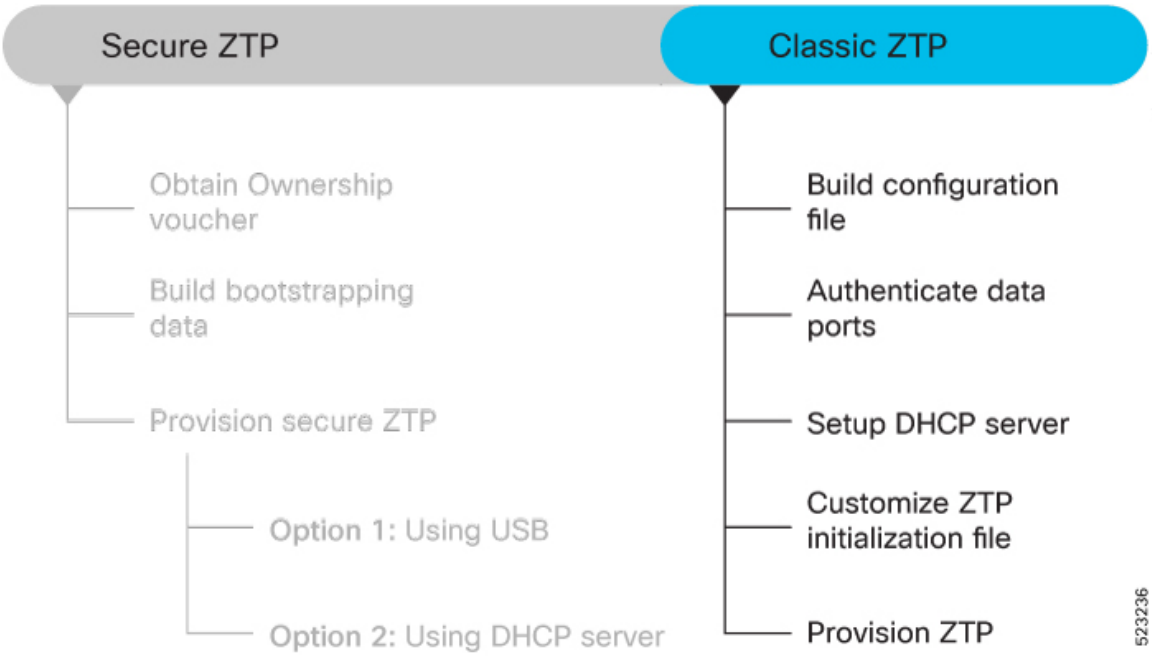
- Using ZTP to integrate IOS-XR with NSO
- Using ZTP to install Puppet

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.
- **Manual Invocation:** Use this method when you want to forcefully initiate ZTP on a fully configured device. To know the detailed steps of manual invocation of ZTP, see Manual Invocation of ZTP, on page 148.
- **ZTP Bootscript**: Use this method when you want to hard code a script to be executed on every boot.

Follow the workflow to understand the tasks involved in provisioning the router using classic ZTP.

Figure 21: Classic ZTP Workflow



This section contains the following topics:

- Deploy Router Using Classic ZTP, on page 117
- Build Configuration File, on page 118
- Limitations of temporary ZTP files, on page 134
- Authenticate Data Ports, on page 135
- Setup DHCP Server, on page 138
- Customize ZTP Initialization File, on page 141
- ZTP file download behavior with libcurl, on page 144
- Zero Touch Provisioning on a Fresh Boot of a Router, on page 144
- Manual Invocation of ZTP, on page 148

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. Currently, ZTP only supports single name-server. When the DHCP server has more than one server address configured, ZTP fails to apply the server configuration.

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

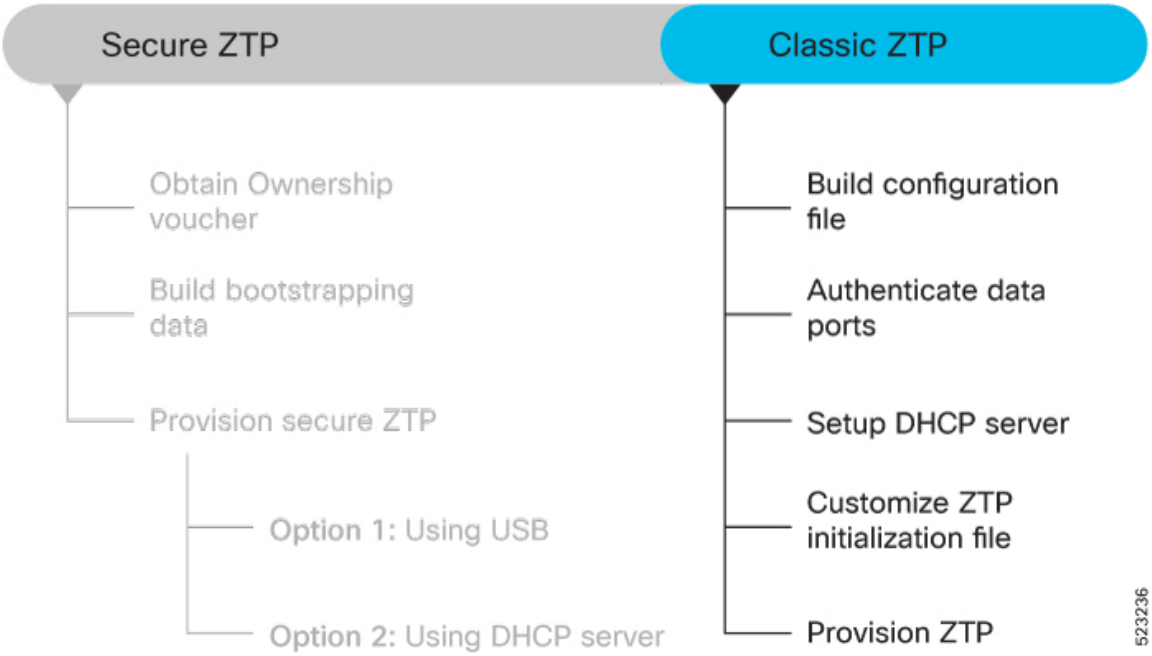
You can initiate ZTP in one of the following ways:

- **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.

- **Manual Invocation:** Use this method when you want to forcefully initiate ZTP on a fully configured device. To know the detailed steps of manual invocation of ZTP, see Manual Invocation of ZTP, on page 148.
- **ZTP Bootscript**: Use this method when you want to hard code a script to be executed on every boot.

Follow the workflow to understand the tasks involved in provisioning the router using classic ZTP.

Figure 22: Classic ZTP Workflow



This section contains the following topics:

Build Configuration File

Table 28: Feature History Table

Feature Name	Release Information	Feature Description
Build Configuration File	Release 25.4.1	Introduced in this release on: Fixed Systems (8010 [ASIC: A100])(select variants only*) *This feature is now supported on: • 8011-12G12X4Y-A • 8011-12G12X4Y-D

Feature Name	Release Information	Feature Description
Build Configuration File	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100, K100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) Based on the business need, you can use a configuration or script file to initiate the ZTP process. *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 8712-MOD-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM

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

Attention 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, #! /usr/bin/python, or #! /usr/bin/python3*.

Note Update your ZTP scripts from python2 to python3, which is supported from Cisco IOS XR Release 24.1.1. You can also use python2-based ZTP scripts until Cisco IOS XR Release 25.2.1.

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

Note We recommend that you don't 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 which 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 don't 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's 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.

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
!
```

You can also use a script file to initiate the ZTP process. 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 you choose. Build your ZTP script with either shell and Python.

Note ZTP does not have its own Python implementation. Instead, ZTP uses the default Python version and libraries available on the device. Python versions and supported libraries vary across Cisco IOS XR software versions and platforms (PID). Make sure that your script works correctly on the specific platform (PID) and required image version before using it in ZTP.

ZTP includes a set of CLI commands and a set of shell utilities that can be used within the user script. ZTP includes a set of shell utilities that can be sourced within the user script. The `ztp_helper.sh` is a shell script that can be sourced by the user script. This script provides simple utilities to access XR functionalities. For information on helper APIs, see the Github repository.

The following shows the sample script in python3.

```
[testenv:~]$ python3 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,',
```

The XML-encoded YANG configuration that follows shows various network settings including:

- Basic setup, including line configuration (TTY, VTY)
- User setup such as System Utilities
- Network configurations such as, domain service, Management IP address assignment, NETCONF, IP routing
- Protocol configurations such as, SSH, LLDP, gRPC
- Security (AAA)
- Interface settings (Interface (IF) manager)

```
Router# python3 ztp_XML_test.py
# netconf_client_ztp_lib - version 1.2 #
2021-02-22 13:53:11,587 - DebugZTPLogger - DEBUG - netconf init attempt: 1
Building configuration...
2021-02-22 13:53:18,117 - DebugZTPLogger - DEBUG - Netconf yang agent is up
###################### Netconf response: Current running configuration ######################
<?xml version="1.0"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
 <data>
  <netconf xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-man-xml-ttyagent-cfg>
   /* Enables NETCONF agent over TTY*/
   <agent>
    <tty>
     <enable></enable>
    </tty>
   </agent>
  </netconf>
<lldp xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-ethernet-lldp-cfg>
 /*Enables and configures global LLDP subcommands*/
  <enable>true</enable>
</lldp>
 <ip-domain xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-ip-domain-cfg>
```

```
  /*Configures domain service related commands*/
  <vrfs>
   <vrf>
    <vrf-name>default</vrf-name>
    <name>cisco.lab</name>
    <servers>
     <server>
      <order>0</order>
      <server-address>5.38.4.246</server-address>
     </server>
    </servers>
   </vrf>
  </vrfs>
</ip-domain>
<interface-configurations xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg>
 /*Configures Interfaces and controls their activation and deactivation*/
  <interface-configuration>
   <active>act</active>
   <interface-name>HundredGigE0/0/0/14</interface-name>
   <shutdown></shutdown>
  </interface-configuration>
  <interface-configuration>
   <active>act</active>
   <interface-name>MgmtEth0/RP0/CPU0/0</interface-name>
   <ipv4-network xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-ipv4-io-cfg>
    /*Configures IPv4 Interface input and output settings on the device*/
    <addresses>
     <primary>
      <address>5.38.9.29</address>
      <netmask>255.255.0.0</netmask>
     </primary>
    </addresses>
   </ipv4-network>
  </interface-configuration>
  <interface-configuration>
   <active>act</active>
   <interface-name>FourHundredGigE0/0/0/0</interface-name>
   <shutdown></shutdown>
  </interface-configuration>
</interface-configurations>
<netconf-yang xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-man-netconf-cfg>
 /*Configures Network Configuration Protocol (NETCONF) commands*/
  <agent>
   <ssh>
    <enable></enable>
   </ssh>
  </agent>
</netconf-yang>
<tty xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-tty-server-cfg>
  <tty-lines>
   <tty-line>
    <name>default</name>
    <exec>
     <timeout>
      <minutes>0</minutes>
      <seconds>0</seconds>
     </timeout>
    </exec>
    <general>
     <absolute-timeout>0</absolute-timeout>
    </general>
   </tty-line>
  </tty-lines>
</tty>
```

```
<host-names xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-shellutil-cfg>
 /*Configures various system utilities related to the shell environment
of the system such as Hostname, Time zone, Prompt, Environmental variable configurations.*/

  <host-name>SF-1</host-name>
</host-names>
<grpc xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-man-ems-cfg>
  <port>57400</port>
  <no-tls></no-tls>
  <enable></enable>
</grpc>
<aaa xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-lib-cfg>
 /*Configures AAA (Authentication, Authorization, and Accounting) settings on the device*/
  <usernames xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-locald-cfg>
   <username>
    <ordering-index>0</ordering-index>
    <name>cafyauto</name>
    <usergroup-under-usernames>
     <usergroup-under-username>
      <name>root-lr</name>
     </usergroup-under-username>
     <usergroup-under-username>
      <name>cisco-support</name>
     </usergroup-under-username>
    </usergroup-under-usernames>
    <secret>
     <type>type10</type>
     <secret10>$6$iY.Zo/7E7RIG5o/.$PH1YegMZiHsiRDTxKOjKQ0i8rd4n
s2vHMHEmQrsMQrrtNTlj/gcBEQRXj3WDR8bAv0rWzz3aGdElteshHYXXR1</secret10>
    </secret>
   </username>
  </usernames>
  <accountings>
   <accounting>
    <type>commands</type>
    <listname>default</listname>
    <type-xr>start-stop</type-xr>
    <method1>local</method1>
   </accounting>
  </accountings>
</aaa>
<ssh xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-crypto-ssh-cfg>
 /*Configures the Secure Shell (SSH) settings on a device such as Encryption, Authentication,
 Session Management*/
  <server>
   <timeout>120</timeout>
   <rate-limit>600</rate-limit>
   <session-limit>110</session-limit>
   <v2></v2>
   <vrf-table>
    <vrf>
     <vrf-name>default</vrf-name>
     <enable></enable>
    </vrf>
   </vrf-table>
   <netconf>830</netconf>
   <netconf-vrf-table>
    <vrf>
     <vrf-name>default</vrf-name>
     <enable></enable>
    </vrf>
   </netconf-vrf-table>
  </server>
</ssh>
```

```
<router-static xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-ip-static-cfg>
/*Configures static IP routing on network devices*/
  <default-vrf>
   <address-family>
    <vrfipv4>
     <vrf-unicast>
      <vrf-prefixes>
       <vrf-prefix>
        <prefix>0.0.0.0</prefix>
        <prefix-length>0</prefix-length>
        <vrf-route>
         <vrf-next-hop-table>
          <vrf-next-hop-next-hop-address>
           <next-hop-address>5.38.0.1</next-hop-address>
          </vrf-next-hop-next-hop-address>
         </vrf-next-hop-table>
        </vrf-route>
       </vrf-prefix>
      </vrf-prefixes>
     </vrf-unicast>
    </vrfipv4>
   </address-family>
  </default-vrf>
</router-static>
<vty xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-tty-vty-cfg>
/*Configures virtual terminal lines (VTY lines) to access a device through SSH or TTY
protocols remotely.*/
  <vty-pools>
   <vty-pool>
    <pool-name>cafyauto</pool-name>
    <first-vty>5</first-vty>
    <last-vty>99</last-vty>
    <line-template>cafyauto</line-template>
   </vty-pool>
  </vty-pools>
</vty>
<netconf-yang xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-netconf-yang-cfg>
/*Configures the Network Configuration Protocol (NETCONF) settings and Yet Another Next
Generation (YANG) data modelling.*/
  <agent>
   <ssh/>
  </agent>
</netconf-yang>
<vty-pool xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-vty-pool-cfg>
 /*Configures virtual terminal (VTY) lines on large number of network devices*/
  <pools>
   <pool>
    <pool-name>cafyauto</pool-name>
    <first-vty-number>5</first-vty-number>
    <last-vty-number>99</last-vty-number>
    <line-template>cafyauto</line-template>
   </pool>
  </pools>
</vty-pool>
<interfaces xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-interface-cfg>
 /*Configures Interface settings such as interface, security, and performance on a device*/

  <interface>
   <interface-name>HundredGigE0/0/0/14</interface-name>
   <shutdown/>
  </interface>
  <interface>
   <interface-name>MgmtEth0/RP0/CPU0/0</interface-name>
   <ipv4>
```

```
    <addresses xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-if-ip-address-cfg>
     /*Configures IP address settings on network interfaces of a device.*/
     <address>
      <address>5.38.9.29</address>
      <netmask>255.255.0.0</netmask>
     </address>
    </addresses>
   </ipv4>
  </interface>
</interfaces>
<lldp xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-lldp-cfg/>
/*Configures the Link Layer Discovery Protocol (LLDP) settings on a network device.*/
<domain xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-domain-cfg>
 /*Configures domain settings on the device*/
  <name>cisco.lab</name>
  <name-servers>
   <name-server>
    <order>0</order>
    <address>5.38.4.246</address>
   </name-server>
  </name-servers>
</domain>
<xr-xml xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-xml-agent-cfg>
/*Configures XML agent settings such as Data formatting, Network Management, and Secure
transport layers on the router.*/
  <agent>
   <ssl/>
   <tty/>
   <enable/>
  </agent>
</xr-xml>
<netconf xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-xml-agent-cfg>
  <agent>
   <tty/>
  </agent>
</netconf>
<router xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-router-static-cfg>
 /*Configures the static routing settings on network devices*/
  <static>
   <address-family>
    <ipv4>
     <unicast>
      <prefixes>
       <prefix>
        <prefix-address>0.0.0.0</prefix-address>
        <prefix-length>0</prefix-length>
        <nexthop-addresses>
         <nexthop-address>
          <address>5.38.0.1</address>
         </nexthop-address>
        </nexthop-addresses>
       </prefix>
      </prefixes>
     </unicast>
    </ipv4>
   </address-family>
  </static>
</router>
<ssh xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-ssh-cfg>
 /*Configures the Secure Shell (SSH) settings such as secure remote access,
Encryption, and security on a network device.*/
  <timeout>120</timeout>
  <server>
   <rate-limit>600</rate-limit>
```

```
   <session-limit>110</session-limit>
   <v2/>
   <vrfs>
    <vrf>
     <vrf-name>default</vrf-name>
    </vrf>
   </vrfs>
   <netconf>
    <port>830</port>
    <vrfs>
     <vrf>
      <vrf-name>default</vrf-name>
     </vrf>
    </vrfs>
   </netconf>
  </server>
</ssh>
<grpc xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-grpc-cfg>
  /*Configures the gRPC (Google Remote Procedure Call) on a network device*/
  <port>57400</port>
  <no-tls></no-tls>
</grpc>
<hostname xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-hostname-cfg>
 /*Configures the hostname on a network device*/
  <system-network-name>SF-1</system-network-name>
</hostname>
<aaa xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-aaa-cfg>
  <usernames xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-aaa-task-user-cfg>
   /*Configures the AAA (Authentication, Authorization, and Accounting) parameters on a
network device*/
   <username>
    <ordering-index>0</ordering-index>
    <name>cafyauto</name>
    <group>
     <root-lr/>
     <cisco-support/>
    </group>
    <secret>
     <ten>$6$iY.Zo/7E7RIG5o/.$PH1YegMZiHsiRDTxKOjKQ0i8rd4ns2vHMHEmQrsMQrrtNTlj
/gcBEQRXj3WDR8bAv0rWzz3aGdElteshHYXXR1</ten>
    </secret>
   </username>
  </usernames>
  <accounting>
   <commands>
    <accounting-list>
     <list-name>default</list-name>
     <start-stop/>
     <local/>
    </accounting-list>
   </commands>
  </accounting>
 </aaa>
 <line xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-line-cfg>
   <default>
    <exec-timeout xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-line-exec-timeout-cfg>
     /*Configures the exec timeout settings on network devices for the amount of time that
 the software waits for user to input after the last key has been pressed */
     <timeout-in-minutes>0</timeout-in-minutes>
     <timeout-in-seconds>0</timeout-in-seconds>
    </exec-timeout>
    <absolute-timeout
xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-um-line-general-cfg>0</absolute-timeout>
    /*Configures line settings on network devices*/
```

```
   </default>
 </line>
 </data>
</rpc-reply>
```

Configuration Data Using JSON-encoded YANG Information

Table 29: Feature History Table

Feature Name	Release Information	Feature Description
Configuration Data Using JSON-encoded YANG Information	Release 24.1.1	We now support JSON-encoded data for YANG data instances of configuration for your router. Previously, we supported only XML-encoded data for YANG data instances.

The JSON-encoded YANG configuration that follows shows various network settings including:

- Basic setup, including line configuration (TTY, VTY)
- User setup such as System Utilities
- Network configurations such as, domain service, Management IP address assignment, NETCONF, IP routing
- Protocol configurations such as, SSH, LLDP, gRPC
- Security (AAA)
- Interface settings (Interface (IF) manager)

```
Router# python3 ztp_json_test.py
# netconf_client_ztp_lib - version 1.2 #
2021-02-22 13:53:11,587 - DebugZTPLogger - DEBUG - netconf init attempt: 1
Building configuration...
2021-02-22 13:53:18,117 - DebugZTPLogger - DEBUG - Netconf yang agent is up
###################### Netconf response: Current running configuration ######################
{
    "updates": [
        {
 /* Cisco-IOS-XR-lib-mpp-cfg: Configures library files for managing control plane services.*/


             "path":
"Cisco-IOS-XR-lib-mpp-cfg:control-plane/management-plane-protection/inband/interface-selection/all-interfaces/netconf-protocol/peer-class/peer-v4/peers/peer[address=10.10.10.1]",

            "value": {
                "address": "10.10.10.1"
            }
        },
        {
 /* Cisco-IOS-XR-lib-mpp-cfg: Configures library files for managing control plane services*/

            "path":
"Cisco-IOS-XR-lib-mpp-cfg:control-plane/management-plane-protection/inband/interface-selection/all-interfaces/netconf-protocol/peer-class/peer-v4/peer-prefixes/peer-prefix[address-prefix=10.10.10.0/24]",

            "value": {
                "address-prefix": "10.10.10.0/24"
```

```
            }
        },
        {
/* Cisco-IOS-XR-tty-management-cfg: Configures settings related to managing terminal lines
such as setting session timeouts, access control lists (ACLs)*/

            "path": "Cisco-IOS-XR-tty-server-cfg:tty/tty-lines/tty-line[name=console]",
            "value": {
                "name": "console",
                "exec": {
                    "timeout": {
                        "minutes": "0",
                        "seconds": "0"
                    }
                },
                "general": {
                    "absolute-timeout": "0"
                },
                "Cisco-IOS-XR-tty-management-cfg:connection": {
                    "session-timeout": {
                        "timeout": "0",
                        "direction": "in"
                    }
                }
            }
        },
        {
            "path": "Cisco-IOS-XR-tty-server-cfg:tty/tty-lines/tty-line[name=default]",
            "value": {
                "name": "default",
                "exec": {
                    "timeout": {
                        "minutes": "0",
                        "seconds": "0"
                    }
                },
                "general": {
                    "absolute-timeout": "0"
                },
                "Cisco-IOS-XR-tty-management-cfg:connection": {
                    "session-timeout": {
                        "timeout": "0",
                        "direction": "in"
                    }
                }
            }
        },
        {
/* Cisco-IOS-XR-aaa-lib-cfg: Configures AAA (Authentication, Authorization, and Accounting)
services.*/
            "path":
"Cisco-IOS-XR-aaa-lib-cfg:aaa/Cisco-IOS-XR-aaa-locald-cfg:usernames/username[ordering-index=0][name=cafyauto]",

            "value": {
                "ordering-index": "0",
                "name": "cafyauto",
                "usergroup-under-usernames": {
                    "usergroup-under-username": [
                        {
                            "name": "root-lr"
                        },
                        {
                            "name": "cisco-support"
                        }
```

```
                    ]
                },
                "secret": {
                    "type": "type10",
                    "secret10":
"$6$UTr4x1rhFzPoDx1.$BM7lVnhx10ipmGgVWhxb/eR4cI/x1xWjv95f1ZfCrSxMpkXMH2FxxHKOtBXaqG34A8hMoexOUfStpBwda09j/."

                }
            }
        },
        {
 /*Cisco-IOS-XR-ip-tcp-cfg: Configures library files for managing TCP/IP settings.*/
            "path":
"Cisco-IOS-XR-ip-tcp-cfg:ip/cinetd/services/telnet/vrfs/vrf[vrf-name=default]",
            "value": {
                "vrf-name": "default",
                "ipv4": {
                    "tcp": {
                        "maximum-server": "100"
                    }
                },
                "ipv6": {
                    "tcp": {
                        "maximum-server": "100"
                    }
                }
            }
        },
        {
            "path":
"Cisco-IOS-XR-ip-tcp-cfg:ip/cinetd/services/telnet/vrfs/vrf[vrf-name=genericstring]",
            "value": {
                "vrf-name": "genericstring",
                "ipv4": {
                    "tcp": {
                        "maximum-server": "16"
                    }
                },
                "ipv6": {
                    "tcp": {
                        "maximum-server": "16"
                    }
                }
            }
        },
        {
 /*Cisco-IOS-XR-linux-xlnc-cfg:Configures Linux environment (Xlnc) settings.*/
            "path":
"Cisco-IOS-XR-linux-xlnc-cfg:linux-networking/vrf-names/vrf-name[vrf-name=default]",
            "value": {
                "vrf-name": "default",
                "address-families": {
                    "address-family": [
                        {
                            "address-family": "ipv4",
                            "default-route": {
                                "software-forwarding": {}
                            },
                            "source-hint": {
                                "default-route-source-hint": {
                                    "interface-name": "MgmtEth0/RP0/CPU0/0"
                                }
                            }
                        }
```

```
                    ]
                }
            }
        },
        {
 /*Cisco-IOS-XR-ifmgr-cfg: Configures Interfaces and controls their activation and
deactivation*/
            "path":
"Cisco-IOS-XR-ifmgr-cfg:interface-configurations/interface-configuration[active=act][interface-name=FortyGigE0/0/1/1]",

            "value": {
                "active": "act",
                "interface-name": "FortyGigE0/0/1/1",
                "shutdown": {}
            }
        },
        {
            "path":
"Cisco-IOS-XR-ifmgr-cfg:interface-configurations/interface-configuration[active=act][interface-name=HundredGigE0/0/1/0]",

            "value": {
                "active": "act",
                "interface-name": "HundredGigE0/0/1/0",
                "shutdown": {}
            }
        },
        {
            "path":
"Cisco-IOS-XR-ifmgr-cfg:interface-configurations/interface-configuration[active=act][interface-name=HundredGigE0/0/1/2]",

            "value": {
                "active": "act",
                "interface-name": "HundredGigE0/0/1/2",
                "shutdown": {}
            }
        },
        {
            "path":
"Cisco-IOS-XR-ifmgr-cfg:interface-configurations/interface-configuration[active=act][interface-name=HundredGigE0/0/1/3]",

            "value": {
                "active": "act",
                "interface-name": "HundredGigE0/0/1/3",
                "shutdown": {}
            }
        },
        {
            "path":
"Cisco-IOS-XR-ifmgr-cfg:interface-configurations/interface-configuration[active=act][interface-name=MgmtEth0/RP0/CPU0/0]",

            "value": {
                "active": "act",
                "interface-name": "MgmtEth0/RP0/CPU0/0",
                "Cisco-IOS-XR-ipv4-io-cfg:ipv4-network": {
                    "addresses": {
                        "primary": {
                            "address": "10.105.247.150",
                            "netmask": "255.255.255.0"
                        }
                    }
                }
            }
        },
        {
```

```
            "path":
"Cisco-IOS-XR-ifmgr-cfg:interface-configurations/interface-configuration[active=act][interface-name=GigabitEthernet0/0/0/30]",

            "value": {
                "active": "act",
                "interface-name": "GigabitEthernet0/0/0/30",
                "shutdown": {}
            }
        },
        {
            "path":
"Cisco-IOS-XR-ifmgr-cfg:interface-configurations/interface-configuration[active=act][interface-name=GigabitEthernet0/0/0/34]",

            "value": {
                "active": "act",
                "interface-name": "GigabitEthernet0/0/0/34",
                "shutdown": {}
            }
        },
        {
            "path":
"Cisco-IOS-XR-ifmgr-cfg:interface-configurations/interface-configuration[active=act][interface-name=GigabitEthernet0/0/0/35]",

            "value": {
                "active": "act",
                "interface-name": "GigabitEthernet0/0/0/35",
                "shutdown": {}
            }
        },
        {
 /* Cisco-IOS-XR-man-ems-cfg: configures EMS (Element management System) for real time
network event detection.*/

            "path": "Cisco-IOS-XR-man-ems-cfg:grpc",
            "value": {
                "port": "57400",
                "no-tls": {},
                "enable": {}
            }
        },
        {
 /* Cisco-IOS-XR-man-netconf-cfg: Configures NETCONF (Network Configuration Protocol).*/


            "path": "Cisco-IOS-XR-man-netconf-cfg:netconf-yang/agent/ssh",
            "value": {
                "enable": {}
            }
        },
        {
 /* Cisco-IOS-XR-shellutil-cfg: Configures shell utilities in Linux.*/
            "path": "Cisco-IOS-XR-shellutil-cfg:host-names",
            "value": {
                "host-name": "150"
            }
        },
        {
            "path": "Cisco-IOS-XR-crypto-ssh-cfg:ssh/server",
            "value": {
                "v2": {}
            }
        },
        {
 /* Cisco-IOS-XR-crypto-ssh-cfg: Configures the Secure Shell (SSH) settings on a device
```

```
such as Encryption, Authentication, Session Management.*/
          "path": "Cisco-IOS-XR-crypto-ssh-cfg:ssh/server/vrf-table/vrf[vrf-name=default]",

            "value": {
                "vrf-name": "default",
                "enable": {}
            }
        },
        {
            "path":
"Cisco-IOS-XR-crypto-ssh-cfg:ssh/server/netconf-vrf-table/vrf[vrf-name=default]",
            "value": {
                "vrf-name": "default",
                "enable": {}
            }
        },
        {
            "path":
"Cisco-IOS-XR-ip-static-cfg:router-static/default-vrf/address-family/vrfipv4/vrf-unicast/vrf-prefixes/vrf-prefix[prefix=0.0.0.0][prefix-length=0]",

            "value": {
                "prefix": "0.0.0.0",
                "prefix-length": "0",
                "vrf-route": {
                    "vrf-next-hop-table": {
                        "vrf-next-hop-next-hop-address": [
                            {
                                "next-hop-address": "10.105.247.1"
                            }
                        ]
                    }
                }
            }
        },
        {
 /*Configures virtual terminal lines (VTY lines) to access a device through SSH or TTY
protocols remotely.*/

            "path": "Cisco-IOS-XR-tty-vty-cfg:vty/vty-pools/vty-pool[pool-name=default]",
            "value": {
                "pool-name": "default",
                "first-vty": "0",
                "last-vty": "50"
            }
        },
        {
 /* Cisco-IOS-XR-ethernet-lldp-cfg: Configures global LLDP subcommands.*/
            "path": "Cisco-IOS-XR-ethernet-lldp-cfg:lldp",
            "value": {
                "enable": "true",
                "enable-subintf": "true"
            }
        },
        {
 /*Cisco-IOS-XR-call-home-cfg: Configures Call Home service to contact Cisco or a designated
 server automatically for support in response to certain predefined events.*/
            "path": "Cisco-IOS-XR-call-home-cfg:call-home",
            "value": {
                "active": {},
                "contact-smart-licensing": "true"
            }
        },
        {
            "path":
```

```
"Cisco-IOS-XR-call-home-cfg:call-home/profiles/profile[profile-name=CiscoTAC-1]",
            "value": {
                "profile-name": "CiscoTAC-1",
                "active": {},
                "methods": {
                    "method": [
                        {
                            "method": "email",
                            "enable": "false"
                        },
                        {
                            "method": "http",
                            "enable": "true"
                        }
                    ]
                }
            }
        },
        {
 /* Cisco-IOS-XR-policy-repository-cfg: Configures the policy repository such as QoS,
Routing.*/
            "path":
"Cisco-IOS-XR-policy-repository-cfg:routing-policy/route-policies/route-policy[route-policy-name=PASS_ALL]",

            "value": {
                "route-policy-name": "PASS_ALL",
                "rpl-route-policy": "route-policy
PASS_ALL\n\t\t\t\t\tpass\n\t\t\t\t\tend-policy"
            }
        },
        {
 /* Cisco-IOS-XR-infra-infra-cfg: Configures infrastructure settings such as system logging,
 system clocks, banners, usernames.*/
            "path": "Cisco-IOS-XR-infra-infra-cfg:banners/banner[banner-name=login]",
            "value": {
                "banner-name": "login",
                "banner-text": "TEST"
            }
        }
    ]
}
```

Limitations of temporary ZTP files

During the ZTP (Zero Touch Provisioning) process, temporary files such as */disk0:/ztp/xr_config/interface_list* may remain on disk if the ZTP process does not exit gracefully.

Note For Cisco IOS XR software versions prior to 7.5.1, ZTP used the */disk0:/ztp/xr_config/xr_config_to_revert* file for similar tracking.

```
RP/0/RP0/CPU0:ios#run
[node0_RP0_CPU0:~]$ls /disk0\:/ztp/xr_config/
current  interface_list  xr_config_to_revert

[node0_RP0_CPU0:~]$rm /disk0\:/ztp/xr_config/interface_list
[node0_RP0_CPU0:~]$rm /disk0\:/ztp/xr_config/xr_config_to_revert
```

The ZTP process may exit abruptly under the following conditions:

- The ZTP process crashes.
- The ZTP process is manually killed.
- A reload is triggered from a ZTP script.
- The ZTP process is terminated from a script.
- A device reload or RP switchover occurs while ZTP is running.
- An RP is shut down during the ZTP process and reinserted later.

Before performing a manual upgrade, ensure that these temporary ZTP files (*/disk0:/ztp/xr_config/interface_list* and */disk0:/ztp/xr_config/xr_config_to_revert*) are deleted if they exist on disk. This prevents ZTP from bringing down configured interfaces on the next boot, even when a local username is already configured, before ZTP exits.

Authenticate Data Ports

Table 30: Feature History Table

Feature Name	Release Information	Feature Description
Authenticate Data Ports	Release 25.4.1	Introduced in this release on: Fixed Systems (8010 [ASIC: A100])(select variants only*) *This feature is now supported on: • 8011-12G12X4Y-A • 8011-12G12X4Y-D

Feature Name	Release Information	Feature Description
Authenticate Data Ports	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100, K100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) On fresh boot, the ZTP process is initiated from management ports and may switch to data ports. To validate the connection with the DHCP server, authentication is performed on data ports through DHCP option 43 for IPv4 and option 17 for IPv6. *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 8712-MOD-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM

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 'exr-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 "exr-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-EXR-CONFIG code width 2 length width 2;
option CISCO-EXR-CONFIG.client-identifier code 1 = string;
option CISCO-EXR-CONFIG.authCode code 2 = integer 8;
option CISCO-EXR-CONFIG.md5sum code 3 = string;
option vsio.CISCO-EXR-CONFIG code 9 = encapsulate CISCO-EXR-CONFIG;
subnet6 2001:1451:c632:1::/64{
 range6 2001:1451:c632:1::2 2001:1451:c632:1::9;
 option CISCO-EXR-CONFIG.client-identifier "exr-config";
 option CISCO-EXR-CONFIG.authCode 1;
 #valid md5
 option CISCO-EXR-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";
  }
}
```

Setup DHCP Server

Table 31: Feature History Table

Feature Name	Release Information	Feature Description
Setup DHCP Server	Release 25.4.1	Introduced in this release on: Fixed Systems (8010 [ASIC: A100])(select variants only*) *This feature is now supported on: • 8011-12G12X4Y-A • 8011-12G12X4Y-D
Setup DHCP Server	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100, K100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) For ZTP to operate a valid IPv4 or IPv6 address is required and the DHCP server must send a pointer to the configuration script. *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 8712-MOD-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM

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 (exr-config "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 = "exr-config" "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 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 = "exr-config" "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

Table 32: Feature History Table

Feature Name	Release Information	Feature Description
Saving ZTP Configuration Hashes in Router	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM

Saving ZTP Configuration Hashes in Router	Release 7.9.1	This feature allows you to customize **ztp.ini** file to save the ZTP configuration hashes in the **/disk0:/ztp/** location on the router. The router computes the ZTP configuration hash for all the secure ZTP configurations. The ZTP configurations hashes are useful in detecting and preventing tampering between any secure ZTP configuration and the subsequent reboot.

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 from 0 to 9.



Note Lower the number higher the priority. The value 0 has the highest priority and 9 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
```

- `config_check`: Saves ZTP configuration hashes in the **/disk0:/ztp/** location on the router. By default, the config check is disabled. To enable the config check, add the following entry in the`ztp.ini` file.

```
[Startup]
start:         True
retry_forever: True
config_check: True
```

You can view the ZTP hashes by using the **show ztp log** command as seen below:

```
Router# show ztp log

============================== /var/log/ztp.log ==============================

2023-03-14 12:51:29,251 53612 [Configuration] INF: Provisioning via config replace
2023-03-14 12:51:43,131 53612 [Configuration] INF: Configuration has been applied
2023-03-14 12:51:43,131 53612 [Env ] DEB: cfg::createRefOnConfigCommit: called
```

```
2023-03-14 12:51:44,218 53612 [Env ] DEB: cfg:: Generating hash for File name:
/disk0:/ztp/customer/config.inithash_tmp
2023-03-14 12:51:44,218 53612 [Env ] DEB: cfg::_generateCfgAndSaveHash:: HASH :
c7980cfc23a401bbbf296e3d49c76bf9, type : 1
2023-03-14 12:51:59,715 53612 [Env ] DEB: cfg:: Generating hash for File name:
/disk0:/ztp/customer/config.successhash_tmp
..........
2023-03-14 12:51:59,715 53612 [Env ] DEB: cfg::_generateCfgAndSaveHash:: HASH :
c7980cfc23a401bbbf296e3d49c76bf9, type : 2
2023-03-14 12:52:04,901 53612 [Env ] DEB: cfg::getRefOnSuccess :: called
..........
2023-03-14 12:52:05,403 53612 [Engine ] INF: ZAdmin, current state:active, exit
code:success
2023-03-14 12:52:05,403 53612 [Engine ] INF: ZAdmin, current state:final, exit
code:success: state changed to final
```

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.
```

Prior to Cisco IOS XR Release 7.5.1, the ztp disable command creates a `ztp.ini` file in the `/disk0:/ztp` folder. Starting with Cisco IOS XR Release 7.5.1 and later, the ztp.ini file is created in the

`/disk0:/ztp/internal` folder. This file tracks whether ZTP is disabled, ensuring persistence of the disabled state during upgrades and downgrades.

However, the disabled state of ZTP is not persistent when reimaging is performed between pre-7.5.1 and post-7.5.1 images. This means reimaging across these release boundaries requires reconfiguration to disable ZTP again.

If ZTP was disabled:

- Before upgrading from a pre-7.5.1 image, copy `/disk0:/ztp/ztp.ini` to `/disk0:/ztp/internal`.
- Before downgrading to a pre-7.5.1 image, copy `/disk0:/ztp/internal/ztp.ini` to `/disk0:/ztp`.

ZTP file download behavior with libcurl

Cisco IOS XR ZTP uses the `libcurl` library to download provisioning files such as scripts and software images in releases 24.1.1, 24.2.1, 24.2.2, and 24.2.21. The ZTP process retrieves the download URL from DHCP options and passes it directly as a string to libcurl, which initiates the file transfer.

Key behaviors and considerations for ZTP file downloads with `libcurl` include:

- If the download URL uses HTTPS, `libcurl` automatically attempts the HTTPS connection. ZTP allows downloads from HTTPS servers without needing explicit certificate verification configuration, so HTTPS downloads succeed without extra flags or options.
- Administrators sometimes add the `-k` option to a curl command to allow insecure HTTPS connections. However, ZTP does not interpret `-k` as an option; it treats it as part of the DHCP-provided URL string.

  For example, using a DHCP-provided value like *https://server.example.com/ztp.script -k* causes ZTP to interpret `-k` as part of the path, likely resulting in a 404 error or download failure.
- Do not include command-line options such as `-k` in DHCP-provided ZTP URLs in releases 24.1.1, 24.2.1, 24.2.2, and 24.2.21. Use only a valid file URL, such as `https://server.example.com/ztp.script`.
- If needed, you can disable libcurl file downloads in the ZTP configuration file (`ztp.ini`) in releases 24.1.1, 24.2.1, 24.2.2, and 24.2.21 by adding:

  ```
  [Http]
  libcurl_http_get: False
  ```

  When `libcurl` is disabled, ZTP uses the alternative download method as defined by the framework or falls back to the Linux curl utility.

Following these guidelines ensures that ZTP reliably downloads files using libcurl without errors caused by invalid options in the URL.

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 removable storage device

You can automatically provision a network device using ZTP from a removable storage device such as a USB flash drive. The following configuration types are supported on the removable storage device:

- Device-specific configuration: Located in a folder named after the device's chassis serial number. For example, the path for device-specific configuration is `/USB-path/xr-config/<serial-number>/router-cfg`. For instance, `/USB-path/xr-config/FOC2102R1D0/router-cfg` where `FOC2102R1D0` is the chassis serial number.
- Generic configuration: Located directly in the `xr-config` folder. The sample path is `/USB-path/xr-config/router-cfg`.

High-level workflow of ZTP using a USB flash drive

1. When the device boots, it checks if USB is enabled in the `ztp.ini` file. By default, the USB *fetcher* is enabled and assigned the highest priority.

   The fetcher defines which port ZTP uses to obtain provisioning details. Each port has a fetcher priority defined in the `ztp.ini` file.

2. ZTP checks for the presence of a USB flash drive on the device. If no USB drive is detected, ZTP moves to the next fetcher based on the fetcher priority in the `ztp.ini` file.

3. If a USB flash drive is available, ZTP looks for the provisioning files in the following order:

   a. `<usb-root>/xr-config/<chassis-serial>/ztp_actions.json`

   b. `<usb-root>/xr-config/ztp_actions.json`

   c. `<usb-root>/xr-config/<chassis-serial>/router-cfg`

   d. `<usb-root>/xr-config/router-cfg`

4. If a `ztp_actions.json` file is found, ZTP uses only the configuration path specified inside that JSON file.

   Sample generic configuration:

   ```
   "global": {
       "actions": {
         "config": "xr-config/common/router-cfg"
   ```

5. For device-specific configuration, ZTP looks for the `router-cfg` file inside the folder matching the device's chassis serial number within the `xr-config` folder and applies this device-specific configuration. For example, `/USB-path/xr-config/FOC2102R1D0/router-cfg`.

   Device-specific configuration:

   ```
   "FOC2102R1D0": {
       "actions": {
         "image-install": {
           "boot-image": {
             "os-version": "7.9.1",
             "path": "xr-config/FOC2102R1D0/8000-x64-7.9.1.iso"
           }
         }
       }
     }
   ```

For information about the JSON keys used above, see JSON keys and options in ztp_actions file.

6. If the device-specific configuration is not found, ZTP looks for the generic `router-cfg` file directly inside the `xr-config` folder and applies this generic configuration.
7. If the config path inside `ztp_actions.json` is missing or the file is absent, ZTP retries and does not fall back to `router-cfg`.
8. The device applies the retrieved configuration.
9. The network device is now fully provisioned and operational.

JSON keys and options in ztp_actions.json file

The supported JSON keys and options in `ztp_actions.json` include:

- Top-level keys:
  - `global`: Generic or default action block used when no device-specific block matches.
  - `<chassis-serial>`: Device-specific action block keyed by chassis serial number (e.g., `FOC2102R1D0`), which takes priority over `global`.
  - `actions`: Container for actual ZTP actions.
- Action keys under `actions`:
  - `config`: String path to the XR config file to apply. Required for normal config provisioning. If absent, ZTP retries.
  - `image-install`: Requests image upgrade/install before config apply.
    - `boot-image` sub-container includes:
      - `os-version`: Requested image version string.
      - `path`: Path to ISO/GISO image on USB.
  - `enable-secure-mode`: Boolean (`true` or `false`). When `true`, ZTP enables secure mode and restarts ZTP to continue in Secure ZTP flow. Image or config actions in the same file are ignored. If unsupported, ZTP moves to the next fetcher. If enabling secure mode fails, ZTP exits to avoid undefined state. Default is `false` if absent.

Configure ZTP using USB

Follow these steps to configure ZTP using a USB flash drive:

1. Create the configuration file. See *Build your Configuration File*.



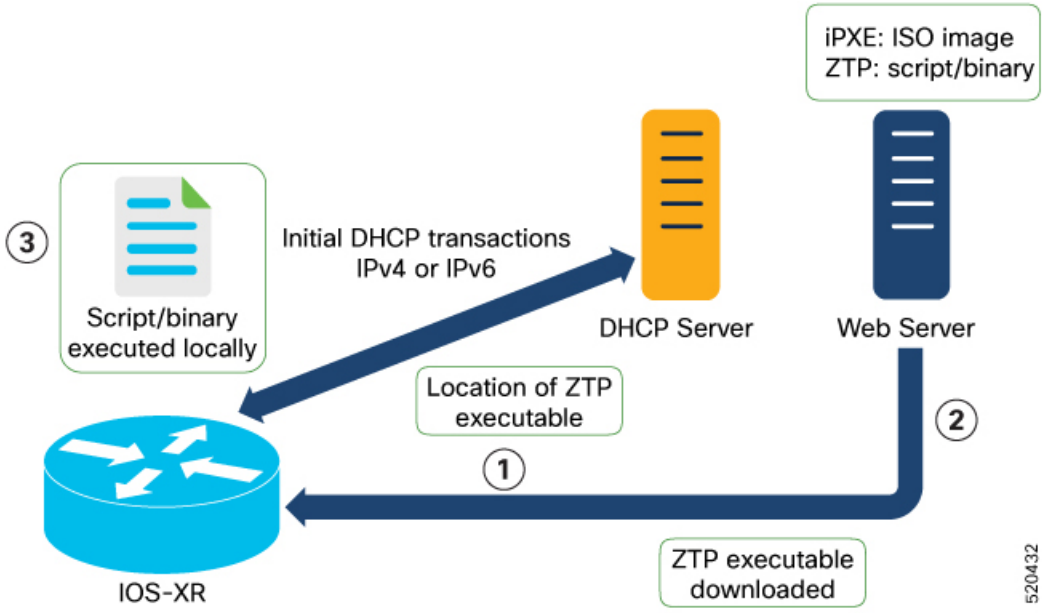
Note When you use a USB flash drive as a source for ZTP, you can't use the script file for provisioning. The script file isn't supported for USB fetcher.

2. Copy the bootstrapping data to the USB flash drive and mount it on the device.

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.

Manual Invocation of ZTP

Procedure

Step 1 Use the **ztp clean** command to remove previous state information from ZTP.

Example:

```
Router#ztp clean
         This would remove all ZTP temporary files.
         Would you like to proceed? [no]: yes
         All ZTP operation files have been removed.
         ZTP logs are present in /var/log/ztp*.log for logrotate.
         Please remove manually if needed.
         If you now wish ZTP to run again from boot, do 'configure terminal/commit replace' followed
 by reload
```

Step 2 Use the **commit replace** command to replace the entire running configuration.

Example:

```
Router#configure terminal
Router(configure)#commit replace
execute "commit replace from configuration mode"
```

Note
- ZTP triggers only if there's no configuration/username.
- The execution of ZTP can be affected by an existing configuration on interface such as DHCP or ACL.
- If the ZTP work flow involves image upgrade, the presence of configuration makes ZTP to exit after reload.

- Existing configuration can cause issue to image upgrade.
- If the existing configuration isn't compatible with the new image upgraded you'll see failures after reimage.

Step 3 Do one of the following steps.

- Use the **reload** command to initiate the ZTP process automatically during system boot on active RP.

```
Router#reload
```

- Use the **ztp initiate** command to initiate ZTP manually.

```
Router#ztp initiate
Initiating ZTP may change your configuration.
Interfaces might be brought up if they are in shutdown state
Would you like to proceed? [no]: yes
ZTP will now run in the background.
```

If all conditions are met ZTP continues with fetch, otherwise exits. The **ztp initiate** command manually initiates ZTP for testing, bypassing all ZTP checks that is followed in the reload method. You shouldn't execute **ztp initiate** on up and running testbed, this command modifies the existing configuration on the router. If the execution of **ztp initiate** fails, make sure of provisioning the device manually.

If you initiate ZTP, using the **ztp initate** command, the ZTP workflow bypasses the exit checks. ZTP will continue running in the background indefinitely. ZTP will exit only if the workflow has completed with a *SUCCESS* or if it is stopped manually. Ensure to stop ZTP using the **ztp terminate** command after initiating it, if ZTP is active/running on the device.



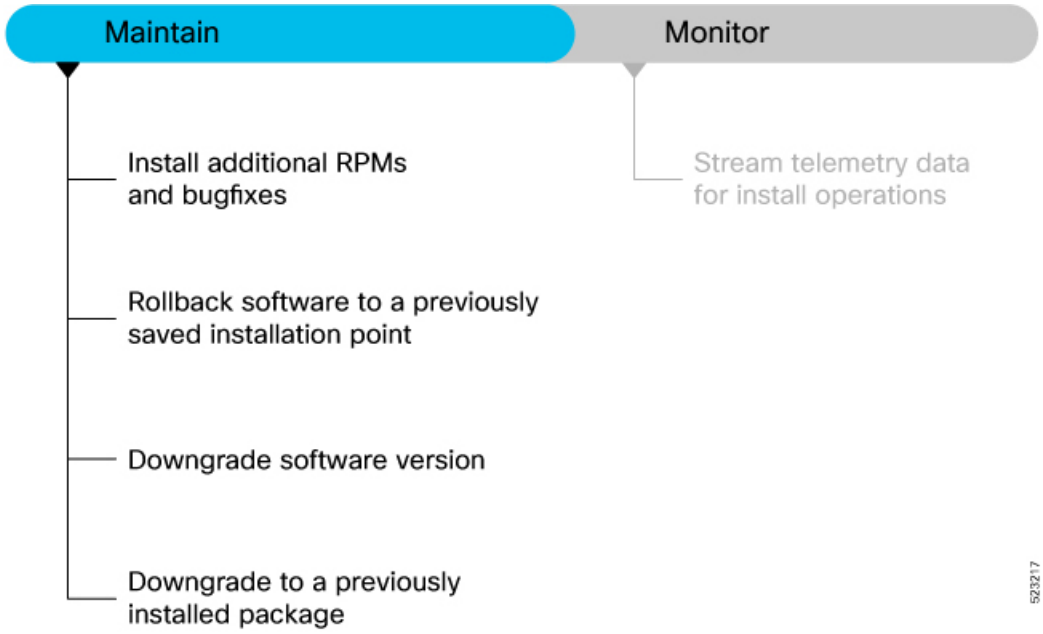
CHAPTER 8

Additional Install Operations

Use the procedures in this section to maintain the router at optimum conditions and monitor the install operation by streaming telemetry data.

The following workflow shows the tasks involved in managing the software:

Figure 23: Workflow to Maintain and Monitor the Software Installation



This section contains the following topics:

- Install Additional RPMs and Bug Fixes, on page 152
- Downgrade Software Version, on page 155
- Remove a bug fix from Cisco IOS XR software, on page 157
- Roll back software to a previously saved installation point , on page 158
- Rollback from SONiC to Cisco IOS XR OS, on page 161
- Stream Telemetry Data for Install Operations, on page 163

Install Additional RPMs and Bug Fixes

You can install individual optional packages when new features are added or software problems are fixed.

Before you begin

Table 33: Feature History Table

Feature Name	Release Information	Feature Description
Install Cisco RPM Directly from TAR File	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM
Install Cisco RPM Directly from TAR File	Release 7.3.1	Cisco RPMs can be installed from one of these sources - repository name, repository URL or a local file path. Previously, to install RPMs from a TAR file, you had to manually extract the TAR file and specify the path to the locally extracted file to install the RPM. This feature extends support to install the RPM directly from the TAR file. Bug fix RPMs are available as TAR files on theSoftware Downloadpage. The RPMs can be installed from the TAR file using CLI or Yang data model.

When you upgrade the Cisco IOS XR software, you can also install or remove optional feature packages (RPMs or bug fixes) *before* applying the changes in the router. You can perform this operation while an atomic change is already in progress. However, all packaging operations before this command are discarded.

You can install the packages from a remote repository or copy the files to the router. If you are using a remote repository, ensure you have created and configured an external repository to store the packages. See the Create Repository to Access Install Files, on page 43 topic.

Download the specific additional RPMs and latest bug fix RPMs as tarballs to the repository. If the bug fix has dependencies, we recommend that you create a bug fix tarball that contains all dependencies.The *README* file in the tarball provides relevant information about the bug fix and identifies any dependencies – for example, whether other bug fix RPMs may be required for a complete fix.

Option 1: Install RPMs Using Command Line Interface

Optional RPMs and bug fixes are available as TAR files on the Software Download page. Starting with Cisco IOS XR Release 7.3.1, you are no longer required to manually extract the RPMs from the TAR file; you can install the bug fix RPM directly from the TAR file.

Procedure

Step 1 Check the available packages in the repository.

Example:

```
Router#show install available

Trying to access repositories...
Package        Architecture        Version            Repository
xr-8000-core    x86_64             7.8.1              remote-repo
xr-core         x86_64             7.8.1              remote-repo
```

Step 2 Install the packages (additional RPMs or bug fixes).

- **Option 1:** Install RPMs without control over reload operation.

  Important
  This option is not applicable when you downgrade or remove RPMs.

  You can either specify a tarfile (with bug fixes or optional packages), or a repository containing the RPMs. Use this command:

  ```
  Router#install source full-path-to-rpm [all]
  ```

  Specify the **all** keyword if you want to install optional packages. Exclude the **all** keyword if you want to upgrade the packages that are currently installed on the system.

  The *full-path-to-rpm* can be one of the following locations based on where you have saved the files.

  - Local path—files located in or under `/var/xr/disk1/`, `/harddisk:/` or `/misc/disk1/`
  - Remote repository or tar file—`ftp://<server>[;<named-vrf>]/<remote_path>`, `https://<server>[;<named-vrf>]/<remote_path>` or `http://<server>[;<named-vrf>]/<remote_path>`

    If you want to add new packages from this source, you must use the **all** keyword:

    ```
    Router#install source full-path-to-rpm all sync
    ```

    Note
    If the remote repository is reachable through a named VRF, you must mention the named VRF in the above commands. For example,

    ```
    Router#install source http://10.105.57.27;vrf1/repoinfra/install_rpms.tar
    ```

    where **vrf1** is the named VRF through which the remote repository is accessible.

The operation adds the RPMs and applies the change via `reload` or `restart` operation, whichever is least impactful based on the update.

- **Option 2:** Install RPMs with control over reload operation.

  Important
  This option is applicable when you downgrade, remove or rollback RPMs.

  a. Install RPMs by providing the RPM name, Cisco bug fix ID (example, CSCab12345) or add packages from a specified source. Use the **install package add** command if you want to add new optional packages, else use the **install package upgrade** command.

  ```
  Router#install package add <pkg1> <pkg2> <pkgn>
  ```

  Or

  ```
  Router#install package upgrade <pkg1> <pkg2> <pkgn>
  ```

  b. Apply the changes.

  ```
  Router#install apply [reload | restart]
  ```

  You can use the `reload` or `restart` options based on the change that is installed. To determine whether a `reload` or `restart` is required, check the output of **show install request** or **show install history last transaction verbose** command. The output indicates the required actions.

  ```
  Router#show install history last transaction verbose
  2023-01-25 05:45:37 UTC    Transaction 87 started
  2023-01-25 05:45:37 UTC      Atomic change 87.1 started
  2023-01-25 05:45:37 UTC      Packaging operation 87.1.1 started
  2023-01-25 05:45:37 UTC    Transaction 87 complete

  Least impactful apply method: process restart
  ```

Step 3 Check the status of the install operation.

Example:

```
Router#show install request
User request: No user requests found
State:        Success
Current activity:    No install operation in progress

The following actions are available:

    install package add
    install package remove
    install package upgrade
    install package downgrade
    install package replace
    install package rollback
    install replace
    install rollback
    install source
```

Note
Include the keyword `noprompt` in the commands to enable the system to bypass your permission to reload the router.

Step 4 Verify the image and packages are activated successfully.

Example:

```
Router# show install request
User request: install package add xr-mcast
Operation ID: 87.1.1
State: Success
```

Step 5 Commit the transaction.

Example:

```
Router#install commit
```

Option 2: Install RPMs Using YANG Data Model

Use `Cisco-IOS-XR-install-augmented-act.yang` data model to install the RPMs or bug fixes.

Procedure

	Command or Action	Purpose

Step 1	Use the `install-package-replace` RPC on the data model.
	Example:
	`<install-package-replace>`
	`  <source-type>remote</source-type>`
	`  <source>remote-repo</source>`
	`  <file>rpm-file-name</file>`
	`</install-package-replace>`
		If the install operation lists the repository reachable through a VRF, you must add the VRF name for the operation to be successful.
		`<install-package-upgrade xmlns=http://cisco.com/ns/yang/Cisco-IOS-XR-install-augmented-act>`
		`  <source-type>ftp</source-type>`
		`<source>10.105.57.27;vrf1/repoinfra/install_rpms.tar</source>`
		`</install-package-upgrade>`

Downgrade Software Version

Before you begin

Check the FPD status and ensure that all the FPDs are in `CURRENT` state.

```
Router#show hw-module location all fpd
```

If the FPDs are not in `CURRENT` state, upgrade the FPDs.

```
Router#upgrade hw-module location all fpd all
```

After all the FPDs are upgraded, reload the router.

```
Router#reload location all
Proceed with reload? [confirm]
```

After the router reloads, check that all the FPDs are in `CURRENT` state.

Note We do not recommend downgrading the FPDs when you downgrade the system.

For more information on upgrading FPDs, see the Upgrading Field-Programable Device chapter.

Downgrade the current software version to a previous software release in case of an upgrade failure or based on requirement.

Procedure

Step 1 Determine the supported target versions to downgrade from the current version.

Example:

```
Router#show install upgrade-matrix
```

View the hardware or software limitations, and bridging SMUs required for the version downgrade. For more information about checking compatibility between the current and target versions, see Supported upgrade and downgrade releases, on page 30.

Downgrading Packages:

Customers can also downgrade user-specified packages (for example, xr-telnet). This is separate from downgrading the entire XR version, but an ISO for an earlier version of XR is used instead of a newer ISO.

Note
The downgrade of IOS XR from version 7.3.4 to 7.0.14 for systems with Open PID RP can cause route processor BIOS corruption. We recommend that you do not downgrade below version 7.3.16.

Step 2 Back up the file system of the current version for recovery purposes.

Example:

Copy the running configuration to the harddisk: directory on the router:

```
Router#copy running-config harddisk:/running_config-<mmddyyyy>
```

Copy the running configuration to a remote server:

```
Router#scp harddisk:/ running_config user@<ip-address>:<location>
```

Step 3 Download the target version from the Software Download Center.

Step 4 You can either install from the remote repository or copy the ISO image file to the /harddisk: of the router.

Example:

```
Router#scp root@<ip-address>:/<dir>/8000-x64-release.iso harddisk:
```

Step 5 Verify that the MD5 checksum of the copied target file matches with the MD5 value of the source on the Software Download Center.

Example:

```
Router#show md5 file /harddisk:/8000-x64-<target-version>.iso
```

Step 6 Install the base image to downgrade the system.

- **Option 1:** Install ISO without control over reload timing.

```
Router#install replace /harddisk:/8000-x64-release.iso
```

The image is installed, the changes are applied through a reload or a restart of the system, and commits the changes. However, you do not have control over the timing of the reload or restart —these occur as soon as the package operation completes and the system is ready.

If you want to control when your system reloads (management of a network outage), we recommend that you schedule a downgrade window and perform an **install replace** operation, letting the system reload without intervention.

- **Option 2:** Install ISO with control over reload timing.

  a. Install the image.

  ```
  Router#install package replace /harddisk:/8000-x64-release.iso
  ```

  b. Apply the changes.

  ```
  Router#install apply [reload | restart]
  ```

  You can use either the `reload` or `restart` options based on the file that is installed. To determine whether a `reload` or `restart` is required, check the output of **show install request** command. The output indicates the required actions.

Step 7 After the base image is downgraded, install the additional packages. For more information, see Install Additional RPMs and Bug Fixes, on page 152.

During an install operation, if the system reboots unexpectedly or an apply by reload results in the system failing to boot, it automatically recovers to its software state before the current transaction.

Remove a bug fix from Cisco IOS XR software

You can remove a bug fix that is causing issues or is no longer needed.

Before you begin

Warning Bug fixes that are prerequisites for other fixes can only be removed if the dependent bug fixes are removed.

Procedure

Use one of the following options to remove a bug fix from your IOS XR software version:

- If the bug fix to be removed was the most recently installed, and the device supports the **install rollback** feature, roll back using rollback ID

  a. Identify the rollback ID for the bug fix you want to remove.

  b. Use the rollback process to return your software to a previous stable version using the rollback ID.

- Build a custom Golden ISO (GISO) image

a. Use the `gisobuild.py` tool to create a custom GISO image that excludes the unwanted bug fix and includes any required fixes.

b. Install the custom GISO image to update your system.

The bug fix is removed from your system using the selected method.

Roll back software to a previously saved installation point

You can roll your system software back to a previous version. This could be used to discard an ongoing install operation, or to undo an install operation that has already been committed. After each commit operation, the system saves a record of the committed software packages. Each record is a restoration point, and is assigned a unique ID. This ID is known as a rollback ID. You can use the rollback ID to roll back the software to a restoration point associated with this ID. Up to 900MB of space is allowed for rollback points, instead of a specific number of rollback points.

Table 34: Feature History Table

Feature Name	Release Information	Feature Description
Rollback changes with simplified install process	Release 25.3.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q200, P100], 8700 [ASIC: P100, K100], 8010 [ASIC: A100]); Centralized Systems (8600 [ASIC:Q200]) ; Modular Systems (8800 [LC ASIC: Q100, Q200, P100]) With the simplified install process that enhances upgrade robustness and scalability between IOS XR versions, rollback packages are no longer available after a release upgrade. You can continue to use the **install rollback** command to reverse operations within the same XR release, and existing rollback commands remain available for reverting operations involving multiple SMUs or optional packages.

Feature Name	Release Information	Feature Description
Roll back to a previously saved installation point	Release 7.4.1	A maximum of 900MB of disk space is used to store all rollback points. You can delete older rollback points to make space for newer or smaller rollback points such as, SMUs. This feature introduces the following commands: • **install commit description**—sets the description for a rollback ID • **install label id** *id* **description** *description*—adds a new description or overwrites the description of the specified rollback ID. The following command is updated to include the keyword, **oldest** *n*: • **clear install rollback oldest** *n*—deletes the specified number of rollback points

Note

- Use rollback ID 0 to roll back to the software that was present after the system booted for the first time.
- If you commit an install transaction using the **install commit** command, the GISO ZTP configuration is saved along with the rest of the software changes. This means that if you use the **install rollback** or **install package rollback** command to revert the software to the state of a previous transaction, the GISO ZTP configuration is also reverted to its previous state. To undo this install operation, as well as the change in GISO ZTP configuration, use the **install package abort** command. If there is no GISO ZTP configuration saved at the end of the install transaction, the existing GISO ZTP configuration is reverted to the previous state when the rollback operation of that transaction is applied.

Roll back Cisco IOS XR software using a rollback ID

This procedure helps you rollback your software using a rollback ID.

Procedure

Step 1 View the list of available rollback IDs.

Example:

```
Router# show install rollback list-ids
```

Step 2 Explore the main packages that can be installed if you roll the software back to the specific rollback ID.

Example:

```
Router# show install rollback id <id>
```

Step 3 View the relative changes that are made to the currently installed software if it is rolled back to a rollback ID.

Example:

```
Router# show install rollback id <id> changes
```

To perform these tasks using data models, use the `get` RPC on the Cisco-IOS-XR-install-augmented-oper data model.

```
<rpc>
  <get>
   <filter type="subtree">
      <install xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-install-augmented-oper">
      <rollback/>
     </install>
   </filter>
  </get>
</rpc>
```

Step 4 Roll back to the software associated with the specific rollback ID.

Example:

```
Router# install rollback <id> [commit]
```

If you want to apply the change and roll back to the associated rollback ID, commit the change. You can also include the keyword `noprompt` in the command to enable the system to bypass your permission to reload the router.

Attention
This roll back operation installs the previous software and also applies the change automatically. This may reload the router depending on the package that is rolled back.

Alternatively, use the **install package rollback** command to only roll back the package but not apply the changes. You can check whether the router will reload or restart if you apply the change using the **show install history last transaction verbose** command or **show install request** command. Based on the command output, you can take the appropriate action using the **install apply reload | restart** command to either reload or restart the system. Use the **install commi**t command to commit the transaction.

To perform this task using data models, use the `install-rollback` RPC on the Cisco-IOS-XR-install-augmented-oper data model.

```
<rpc>
 <install-rollback xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-install-augmented-act">
  <commit>true</commit>
   <transaction-id>0</transaction-id>
```

```
 </install-rollback>
</rpc>
```

To understand the data model structure and its arguments, see the Access the Install-Related Data Model.

Step 5 Commit the operation.

Example:

```
Router#install commit
```

Rollback from SONiC to Cisco IOS XR OS

This section describes how to rollback from SONiC OS to Cisco IOS XR software on the router.

Before you begin

Complete these prerequisites before you install Cisco IOS XR software on a router running SONiC:

- Ensure all SONiC instance are running with FPD version 0.1.

```
root@sonic#cardevent.py --send CV_FPDPUBLISH --slot all
root@sonic#fpd-util.py --getfpd
1.0.0.13_programed 0.1
1.0.0.3_programed 0.1
1.0.0.5_programed 0.1
1.0.0.33_programed 0.1
```

- Ensure that chassis can access the DHCP or PXE server hosting the IOS XR image.
- Check the BIOS version on RP and LC to ensure that the BIOS version required for IOS XR boot operation is available.

  RP:

```
cisco@sonic#fwutil show status
Chassis    Module    Component    Version    Description
---------  --------  -----------  ---------  --------------------------------
8800-RP              BIOS         1-25       BIOS - Basic Input Output System
                     Aldrin       1.2        Marvell - Aldrin Ethernet switch
                     Aikido       1.35       Aikido - x86 FPGA
                     TAM          2.5        TAM FW - x86
```

  LC:

```
cisco@sonic#fwutil show status
Chassis      Module    Component    Version    Description
-----------  --------  -----------  ---------  --------------------------------
8800-LC-48H            BIOS         1-25       BIOS - Basic Input Output System
                       Aldrin       -1.65535   Marvell - Aldrin Ethernet switch
```

- Copy the IOS XR image to router as `onie-recovery-x86_64-cisco_8000-r0.efi64.pxe` image.

  RP:

```
cisco@sonic#ifconfig eth0 192.0.2.254 netmask 255.255.0.0
```

  Linux:

```
node$:scp 8000-x64-7.10.1.iso cisco@192.0.2.254:/ws/
```

RP:

```
cp /ws/8000-x64-7.10.1.iso
/opt/cisco/var/tftp/onie-recovery-x86_64-cisco_8000-r0.efi64.pxe
```

Procedure

Step 1 Run the migration script.

Example:

```
root@sonic#xrmigration.sh
   INFO: Staging LC found : 1.0.0.3
   INFO: ipxe container start
   INFO: ipxe container service already running
   INFO: override ONIE image with XR image on staging LC
   INFO: Create dummy sonic image as onie-installer.bin on staging LC for SONiC ipxe server
   INFO: XR ethswitch upgrade on all LC
   INFO: Set migration context at staging LC0
   INFO: Set migration context at RP
   Reload all cards in 30 sec
Handling chassis reload scenario...
```

After two reloads, the RP reaches the iPXE server to automatically install the IOS XR image.

Step 2 Reload all line cards.

Example:

```
Router#reload boot media network location LC
```

Step 3 Verify the status of the cards.

Example:

```
Router#show platform
Thu Jun  1 21:38:26.276 UTC
Node              Type                       State                  Config state
--------------------------------------------------------------------------------
0/RP0/CPU0        8800-RP(Active)            IOS XR RUN             NSHUT
0/0/CPU0          88-LC0-36FH                IOS XR RUN             NSHUT
0/1/CPU0          8800-LC-48H                IOS XR RUN             NSHUT
0/5/CPU0          88-LC0-36FH-M              IOS XR RUN             NSHUT
0/FC0             8808-FC0                   OPERATIONAL            NSHUT
0/FT0             8808-FAN                   OPERATIONAL            NSHUT
0/FT1             8808-FAN                   OPERATIONAL            NSHUT
0/FT2             8808-FAN                   OPERATIONAL            NSHUT
0/FT3             8808-FAN                   OPERATIONAL            NSHUT
0/PT0             8800-HV-TRAY               OPERATIONAL            NSHUT
0/PT1             8800-HV-TRAY               OPERATIONAL            NSHUT
0/PT2             8800-HV-TRAY               OPERATIONAL            NSHUT
```

Step 4 After IOS XR software is installed on both RP and LC, reload all the nodes on the router.

Example:

```
Router#reload location all
```

The OS is migrated from SONiC to Cisco IOS XR software.

Stream Telemetry Data for Install Operations

Table 35: Feature History Table

Feature Name	Release Information	Description
Stream Telemetry Data about Install Operations	Release 7.5.2	You can stream telemetry data for install-related details such as active and committed packages, view the progress of install operations, retrieve the image version, and view the error messages with recovery information when an operation fails.

To stream telemetry data that is related to software installation, you must create subscriptions to the sensor paths in the YANG data models. See *Obtain Data Models for Install Operation* for the list of supported data models. For information about establishing a telemetry session and creating subscriptions, see the *Telemetry Configuration Guide for Cisco 8000 Series Routers*.

Stream Telemetry Data About	Description	YANG Path
Summary of active packages	Data is streamed after a successful **apply** operation. An active package is the software currently running on the system.	`Cisco-IOS-XR-install-oper: install/packages/active/summary`
Summary of committed packages	Data is streamed after a successful **commit** operation. A package that is committed remains active following a system reload.	`Cisco-IOS-XR-install-oper: install/packages/committed/summary`
Status of the last request operation	Data is streamed when starting a new request and also when entering an `idle` state. If the operation has failed, this includes error messages along with recovery state.	`Cisco-IOS-XR-install-oper: install/request`
Image version and GISO label	Data is streamed after a successful **apply** operation.	`Cisco-IOS-XR-install-oper: install/version`
Packaging information	Data is streamed at the start and end of a packaging operation.	`Cisco-IOS-XR-install-augmented-oper: install/history/latest-packaging-operation`
Atomic information	Data is streamed at the start and end of **apply** operation.	`Cisco-IOS-XR-install-augmented-oper: install/history/latest-atomic-change`

Stream Telemetry Data About	Description	YANG Path
Transaction information	Data is streamed at the start, in progress, and end of a **commit** operation. **Note** After a transactional rollback, some of the data such as summary of active packages, image version can change. However, telemetry events are not sent after the reload operation.	`Cisco-IOS-XR-install-augmented-oper:` `install/history/latest-transaction`



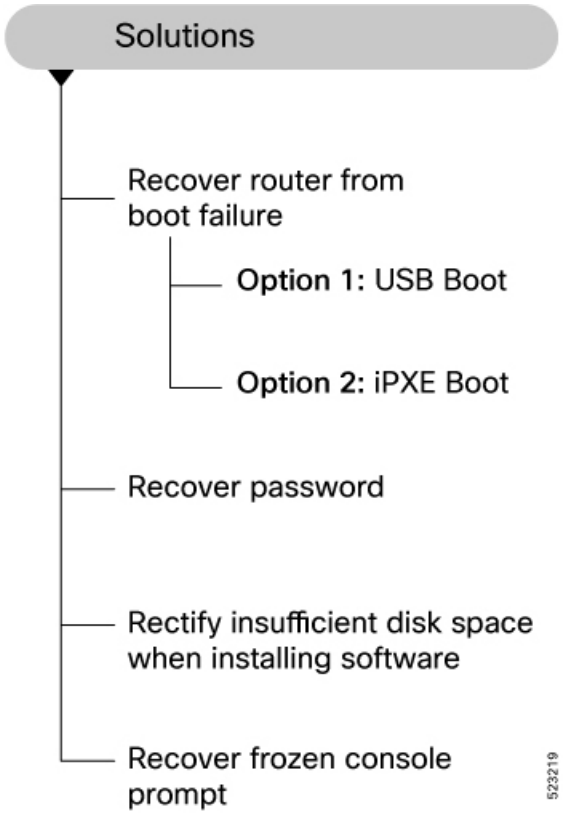
CHAPTER 9

Troubleshoot Router Setup and Upgrade

Use the procedures in this section to troubleshoot router bring-up, software upgrade or downgrade by understanding the problem, probable cause, and the solution.

The following image shows the tasks involved in finding solutions to router setup and upgrade issues:

Figure 24: Solutions to Troubleshoot Software Setup and Upgrade



This section contains the following topics:

- Recover router from boot failure, on page 166
- Recover password, on page 172
- Rectify insufficient disk space when installing software, on page 174
- Recover frozen console prompt, on page 176

Recover router from boot failure

If the command line interface is not accessible, you can recover the router from a boot failure using one of these recovery methods.

Boot the router using USB drive

Problem:

After installing the hardware, you boot the router after connecting to the console port and powering ON the router. The router initiates the boot process using the pre-installed operating system (OS) image. But the router fails to boot, times out or stops responding after the boot process initializes.

Cause:

The router does not boot if an install image is not present on the router or the image is corrupt.

Solution:

Boot the router using a bootable USB flash drive.

The bootable USB flash drive is used to reimage the router during system upgrade or boot the router in case of boot failure. During the USB boot process, the router is re-imaged with the version available on the USB flash drive.

To boot the router using a USB flash drive, you need the following devices:

- A local machine (Windows, Linux, or MAC) with USB Type-A.
- USB flash drive with a storage capacity that is between 8GB (min) and 32 GB (max). USB 2.0 and USB 3.0 are supported.

Note USB Type-C is not supported.

Procedure

Step 1 Create a bootable USB flash drive from your local machine (Windows or MAC):

a) Connect the USB flash drive to your local machine and format it with File Allocation Table (FAT) 32 file system using the Windows Operating System or Apple MAC Disk Utility. Formatting the USB drive to FAT creates addressable sectors that ensures that each piece of information in the file can be found by the computer.

After formatting the USB flash drive, right-click on the USB disk and view the properties.

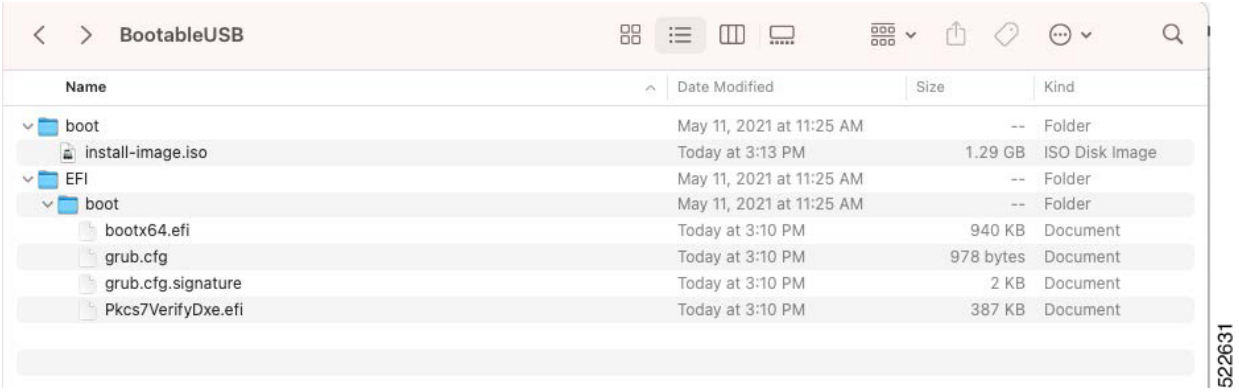
b) On the Software Download page, navigate to the required Cisco IOS XR product and release. The USB boot image is available in the format `<platform>-usb-<version>.zip` compressed file. For example, the USB boot image for Cisco 8000 series routers for release 7.10.1 is `8000-x64-usb-7.10.1.zip` file.

c) Download the compressed USB boot image from the Software Download page to your host computer.

d) Verify that the copy operation is successful. To verify, compare the file size on the Software Download page and the copied file on your computer. You can also verify the MD5 checksum value. This value ensures that the copied file is valid and untampered.

e) Unzip the file to extract the content of the compressed boot file inside the USB flash drive. This converts the USB flash drive to a bootable drive.

Figure 25: Bootable USB Files



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

f) Remove the USB flash drive from your computer.

The USB flash drive is ready to be used as a bootable disk to install and boot the Cisco IOS XR image.

Step 2 Boot the router using the bootable USB flash drive.

a) Use this procedure only on active RP; the standby RP must either be powered OFF or removed from the chassis. After the active RP is installed with images from USB, insert or power ON the standby RP as appropriate.

b) Connect to the console.

c) Insert the USB flash drive in the USB Port Type-A on the router.

Ensure that the router is powered ON. When the USB bootable drive is plugged into an operational router, the device is detected as disk2:. Verify using **show media location all** command.

```
Router#show media location all
Fri Jan 27 08:29:00.808 UTC

Media Info for Location: node0_RP0_CPU0
Partition         Size           Used          Percent        Avail
------------------------------------------------------------------
rootfs:           54.4G          16.5G          30%           38G
data:             77.3G          20.5G          27%           56.8G
disk0:            3.9G           12M            1%            3.6G
/var/lib/docker   6.6G           17M            1%            6.2G
disk2:            15G            6.1G           42%           8.6G
log:              5.3G           572M           12%           4.4G
harddisk:         61G            19G            32%           39G
```

d) View the contents of the USB drive.

Example:

```
Router#dir disk2:
```

e) Initiate the reimage from the USB bootable drive.

Example:

```
Router#reload bootmedia usb noprompt
```

Note
If the router was powered OFF, power ON the router. Press the `Esc` key continuously to pause the boot process and get the RP to the BIOS menu. Use the arrow key and navigate to the `USB Flash Memory` option in the **Boot Manager** menu, and press the `Enter` key. The BIOS GRUB automatically detects the image from the USB flash drive, starts the installation, and displays the progress of the installation operation.

The router reboots after the reimage with new version available in the USB drive. After the installation is complete, the router reboots and enters the prompt to configure the root username and password.

Boot the router using iPXE

Problem:

You connect to the console port and power ON the router. The router initiates the boot process using the pre-installed operating system (OS) image. But the router fails to boot, times out or stops responding after the boot process initializes.

Cause:

The router does not boot if an install image is not present on the router or the image is corrupt.

Solution:

Boot the router using the image from an iPXE server.

iPXE is a pre-boot execution environment that is included in the network card of the management interfaces. It works at the system firmware (UEFI) level of the router. iPXE enables network boot for a router that is offline. The bootloader downloads and installs the ISO image located on an HTTP, FTP, or TFTP server. iPXE boot re-images the router. 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.

Procedure

Step 1 Configure the DHCP server for IPv4, IPv6, or both communication protocols before you use the iPXE boot.

a) Create `dhcpd.conf` file in `/etc/` or `/etc/dhcp` directory. This configuration file stores the network information such as the path to the script, location of the ISO install file, location of the provisioning configuration file, serial number, MAC address of the router. The following example shows a sample `dhcpd.conf` file.

Example:

```
allow bootp;
allow booting;
ddns-update-style interim;
option domain-name "cisco.com";
option time-offset -8;
ignore client-updates;
default-lease-time 21600;
```

```
max-lease-time 43200;
option domain-name-servers <ip-address-server1>, <ip-address-server2>;
log-facility local0;
 :
subnet <subnet> netmask <netmask> {
  option routers <ip-address>;
  option subnet-mask <subnet-mask>;
  next-server <server-addr>;
}
  :
host <hostname> {
  hardware ethernet e4:c7:22:be:10:ba;
  fixed-address <address>;
  filename "http://<address>/<path>/<image.bin>";
```

b) Test the server once the DHCP server is running. For example, for IPv4 protocol:

- Use the MAC address of the router:

Note
Using the `host` statement provides a fixed address that is used for DNS, however, verify that option `77` is set to iPXE in the request. This option is used to provide the boot file to the system when required.

```
host <platform>
{
hardware ethernet <router-mac-address>;
if exists user-class and option user-class = "iPXE" {
 filename = "http://<httpserver-address>/<path-to-image>/<image>";
}
```

Ensure that the above configuration is successful.

- Use the serial number of the router:

```
host <platform>
{
option dhcp-client-identifier "<router-serial-number>";
  filename "http://<IP-address>/<path-to-image>/<image>";
  fixed-address <IP-address>;
}
```

The serial number of the router is derived from the BIOS and is used as an identifier.

Step 2 Recover the router using iPXE boot.

a) Connect to the console.
b) Power ON the router.
c) Press `Esc` key continuously to pause the boot process and get the RP to the BIOS menu.
d) Use the arrow key and navigate to the `Built-in EFI iPXE` option in the **Boot Manager** menu, and press the `Enter` key.

Example:

```
iPXE> ifstat
net0: 00:a0:c9:00:00:00 using i350-b on PCI01:00.0 (closed)
  [Link:up, TX:0 TXE:0 RX:0 RXE:0]
net1: 00:a0:c9:00:00:01 using i350-b on PCI01:00.1 (closed)
  [Link:up, TX:0 TXE:0 RX:0 RXE:0]
net2: 00:a0:c9:00:00:02 using i350-b on PCI01:00.2 (closed)
  [Link:down, TX:0 TXE:0 RX:0 RXE:0]
  [Link status: Down (http://ipxe.org/38086193)]
net3: 00:a0:c9:00:00:03 using i350-b on PCI01:00.3 (closed)
  [Link:down, TX:0 TXE:0 RX:0 RXE:0]
  [Link status: Down (http://ipxe.org/38086193)]
net4: 00:00:00:00:00:04 using dh8900cc on PCI02:00.1 (closed)
```

```
  [Link:down, TX:0 TXE:0 RX:0 RXE:0]
  [Link status: Down (http://ipxe.org/38086193)]
net5: 00:00:00:00:00:05 using dh8900cc on PCI02:00.2 (closed)
  [Link:down, TX:0 TXE:0 RX:0 RXE:0]
  [Link status: Down (http://ipxe.org/38086193)]
net6: 04:62:73:08:57:86 using dh8900cc on PCI02:00.3 (closed)
  [Link:up, TX:0 TXE:0 RX:0 RXE:0]

iPXE> set net6/ip 10.0.0.0
iPXE> set net6/netmask 255.0.0.0
iPXE> set net6/gateway 10.48.42.1
iPXE>
iPXE> ifopen net6

iPXE> ping 10.48.42.1
64 bytes from 10.48.42.1: seq=1
64 bytes from 10.48.42.1: seq=2
Finished: Operation canceled (http://ipxe.org/0b072095)
```

e) Boot the image using one of the following options:

- Option 1: Boot with ISO image. After the reimage is successful, add optional RPMs, bug fixes and update running configuration file.
- Option 2: [Preferred option] Boot with Golden ISO (GISO) image that contains the ISO image, optional RPMs, bug fixes and configuration file. Booting with GISO saves time by eleminating the need to update the files individually.

You must keep the standby RP in the BIOS while installing the image on the active RP.

```
BIOS Ver: 09.19 Date: xx/xx/xxxx 17:02:33

Press <DEL> or <ESC> to enter boot manager.                                    iPXE initialising
devices...ok



iPXE 1.0.0+ (5fbe7) -- Open Source Network Boot Firmware -- http://ipxe.org
Features: DNS HTTP TFTP VLAN EFI ISO9660 NBI Menu
BootMode : 1
Trying net0...
net0: 00:00:01:1c:00:00 using i350-b on PCI01:00.0 (open)
  [Link:up, TX:0 TXE:0 RX:0 RXE:0]
Configuring (net0 00:00:01:1c:00:00)................... ok
net0: 127.0.0.28/255.0.0.0
net0: fe80::2a0:c9ff:fe00:0/64
net1: fe80::2a0:c9ff:fe00:1/64 (inaccessible)
net2: fe80::2a0:c9ff:fe00:2/64 (inaccessible)
net3: fe80::2a0:c9ff:fe00:3/64 (inaccessible)
net4: fe80::200:ff:fe00:4/64 (inaccessible)
net5: fe80::200:ff:fe00:5/64 (inaccessible)
net6: fe80::662:73ff:fe08:1dba/64 (inaccessible)
Next server: 127.0.0.27
Filename: http://127.1.1.27/system_image.iso
http://127.1.1.27/<image>... ok
```

The BIOS GRUB automatically detects the image from the iPXE server, starts the installation, and displays the progress of the installation operation. After the installation is complete, the router reboots and enters the prompt to configure the root username and password.

You can also boot the router from the iPXE server by using the **hw-module location all bootmedia network reload** command.

```
Router# hw-module location all bootmedia network reload
Wed Dec 23 15:29:57.376 UTC
Reload hardware module ? [no,yes]
```

This command configures the router to perform a network-based boot across all modules in the router before a restart. Upon reload, the router attempts to load the operating system image from the specified iPXE server.

Boot the router using install replace reimage

If the standard installation (**install replace**) fails, you can recover the router using the **install replace reimage** command.

Use this command to reimage the router with a fresh copy of the operating system. You can use this command to recover the router from a corrupt state or to upgrade the router to a new major version that requires a fresh install rather than an incremental package upgrade.

```
Router# install replace reimage /harddisk:/8000-64-24.2.11-owner.iso commit
```

Or

```
Router# install replace reimage /harddisk:/8000-64-24.2.11-owner.iso
skip-implicit-owner-packages-checks commit
```

Note The `skip-implicit-owner-packages-checks` parameter is only required if the ISO contains owner or partner RPMs.

Benefits

- You can reimage the router without needing a PXE server setup or USB access.
- The existing XR configuration is retained.
- If the specified iso is a GISO containing its own configuration file, then that configuration is used.
- The **install replace reimage** command resets all partitions and the filesystem on the router, which is useful during error recovery scenarios where you need to clear the existing XR configuration and all partitions from the router.

Limitations

If it is a standard upgrade, it is recommended to use the **install replace** command for upgrading devices rather than the **install replace reimage** command due to these limitations:

- You cannot roll back the operation without an ISO when booting the router using the **install replace reimage** command.
- Booting with the **install replace reimage** command removes all the third party apps installed on the router.

Note Use the **install replace reimage** command only when the standard installation fails.

Recover password

Problem:

Unable to access the router due to incorrect login credentials.

Cause:

A root password is used to login to the router. If you forget this root password, you cannot access the router.

Solution:

If you lose your admin and root user credentials, the router becomes inaccessible. The system can be recovered using a router reimage using iPXE or USB boot. However, this approach is not scalable.

You can use the **system recovery** feature to recover the lost password.

With this feature, the system is recovered without the need to reimage the router. The system is recovered to its initial state with the current running software. The installed software and SMUs are retained after the system is recovered. The process complies with the Cisco Product Security Baseline (PSB) where user data is securely erased before recovering the router. The following data that are generated at run-time are erased:

- XR and admin configuration including the password data
- Cryptographic keys on the disk
- Data on encrypted partition
- Generated core files
- SNMP interface index files
- Third-party application (TPA) software and data
- Files created by the user

Use the following procedure on both RP0 nd RP1 cards on the chassis to recover the password.

Note This procedure is applicable only when you have already enabled the password recovery feature on your router.

```
Router(config)#system recovery
```

Procedure

Step 1 Power ON the router, and press the `ESC` on the RP console to enter the BIOS GRUB menu.

This procedure must be executed on each RP (RP0 and RP1) individually on a modular system.

Step 2 Boot on the standby RP. Press `ESC` key to enter the GRUB (bootstrap program) menu.

Step 3 On the RP0 card console select the **IOS-XR-recovery** option from the GRUB menu and press **Enter**.

Example:

RP0:

Figure 26: IOS XR Recovery Option in GRUB Menu



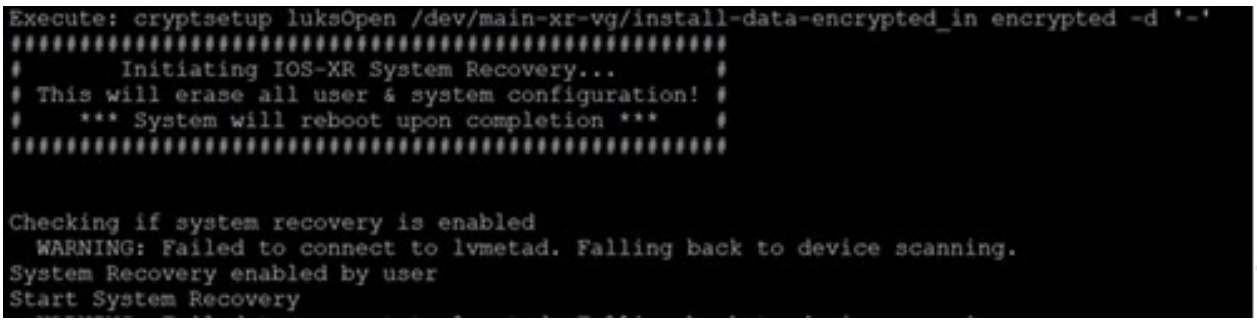
Step 4 Select the **IOS-XR-recovery** option from the GRUB menu and press **Enter** on the RP1 card console when the `Initiating IOS-XR System Recovery...` message is displayed on the RP0 card console.

Note
Do not wait until the RP0 card reaches the `Enter root-system username:` prompt. If you reach this prompt, the RP1 card will reload automatically and exit the BIOS GRUB menu. The RP0 card will boot up as active and the RP1 card will boot up as a standby card post the recovery process.

Example:

RP0:

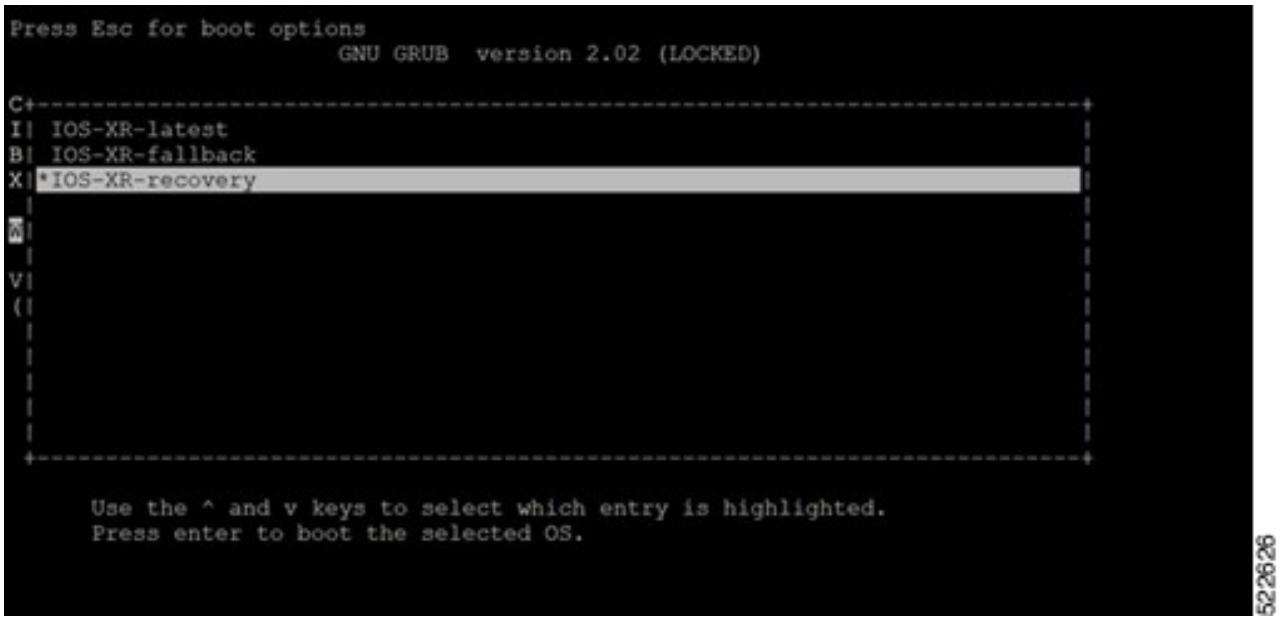
Figure 27: Recovery of RP0



Example:

RP1:

Figure 28: Recovery of RP1



Step 5 On the RP0 card, create a new root user and password. Log in to the router using the new root username and password.

The router boots with the default configuration. Proceed with configuring the router or load a configuration from a backup file if you had already taken a backup. It is recommended to backup data and save the configuration on an external server.

The password recovery procedure is complete.

The option to recover the system using console port is disabled on bootup because all the previous configurations are erased. With this configuration disabled, if you select **IOS-XR-recovery** option from GRUB menu to recover the system, the recovery is skipped. Enable the password recovery feature again using the **system recovery** command.

Rectify insufficient disk space when installing software

Problem:

The software installation terminates with the error:

```
Error on 0/1/CPU0: Insufficient disk space to install packages.
```

Cause:

There is not enough space available for installing the software. Therefore, the installation process terminates with this error syslog:

```
RP/0/RP0/CPU0:Feb 11 05:43:06.449 IST: instorch[415]: %INFRA-INSTALL-4-DISK_SPACE_CHECK_FAILED
: Disk space check failed on nodes: 0/RP0/CPU0. Remove unnecessary files in the rootfs of
the relevant nodes before retrying operation
```

Solution:

There must be enough unused disk space available on the router's *rootfs* location to install the Cisco IOS XR software.

To free up the disk space, perform these steps:

Procedure

Step 1 **show install request**

Example:

```
Router# show install request
Fri Aug 23 09:50:37.418 UTC

User request: install package replace /harddisk:/8000.iso
Operation ID: 1.1.1
State: Failure since 2024-08-23 09:48:46 UTC+00:00
Disk space check failed on nodes: 0/RP0/CPU0. Remove unnecessary files
in rootfs before retrying operation, or remove unnecessary rollback points
using 'clear install rollback id <id>'. If a transaction is currently in
progress, it may need to be ended in order to free up disk space from
removed files or rollback points. Automatically recovered after failure,
ready for next user request.


Location 0/RP0/CPU0
Last event: Error (Insufficient disk space to install packages)
Disk space pre-check failure:
Phase: Veto
Required space: 3099439239B
Available space: 2836786891B
```

Identifies the required disk space.

Step 2 **show media location all**

Example:

```
Router# show media location all
Wed Jan 8 08:29:00.808 UTC

Media Info for Location: node0_RP0_CPU0
Partition                          Size     Used  Percent    Avail
-----------------------------------------------------------------
rootfs:                           54.4G    16.5G      30%      38G
data:                             77.3G    20.5G      27%    56.8G
disk0:                             3.9G      12M       1%     3.6G
/var/lib/docker                    6.6G      17M       1%     6.2G
disk2:                              15G     6.1G      42%     8.6G
log:                               5.3G     572M      12%     4.4G
harddisk:                           61G      19G      32%      39G

Media Info for Location: node0_RP1_CPU0
Partition                          Size     Used  Percent    Avail
-----------------------------------------------------------------
rootfs:                           54.3G    16.5G      30%    37.9G
data:                             77.4G    46.1G      60%    31.4G
disk0:                             3.9G     8.5M       1%     3.6G
/var/lib/docker                    6.6G      19M       1%     6.2G
log:                               5.3G     492M      10%     4.5G
harddisk:                           61G      44G      78%      14G
```

```
Media Info for Location: node0_0_CPU0
Partition                          Size     Used  Percent    Avail
-------------------------------------------------------------------
rootfs:                          54.4G    10.1G      18%    44.4G
data:                            77.3G     1.9G       2%    75.5G
/var/lib/docker                   6.6G      16M       1%     6.2G
disk0:                            3.9G     8.2M       1%     3.6G
harddisk:                          61G     109M       1%      57G
log:                              5.3G     372M       8%     4.6G

Media Info for Location: node0_6_CPU0
Partition                          Size     Used  Percent    Avail
-------------------------------------------------------------------
rootfs:                          54.4G    10.1G      18%    44.4G
data:                            77.3G     1.9G       2%    75.4G
disk0:                            3.9G     8.3M       1%     3.6G
/var/lib/docker                   6.6G      16M       1%     6.2G
harddisk:                          61G     154M       1%      57G
log:                              5.3G     374M       8%     4.6G

Router#
```

Displays the space consumed by the *rootfs:* location.

Use the *commands*[4] from the System Management Command Reference for Cisco 8000 Series Routers to move files from *rootfs* to *harddisk:/* and free up the disk space for the software installation.

Note
The *rootfs:/* includes everything under / that is not mounted elsewhere. For example, */misc/disk1*, */var/log*, */misc/scratch*, and more locations are mounted elsewhere and do not affect the *rootfs* space usage.

The above procedure frees up the required disk space for installation.

What to do next

Retry the install operation once there is enough space.

Recover frozen console prompt

Problem:

The console access is frozen and does not respond. In this state, no output or input characters are displayed on the console.

Cause:

The Priority Flow Control (PFC) functionality is enabed on the console by default. The PFC is also referred to as Class-based Flow Control (CBFC) or Per Priority Pause (PPP) is a mechanism that prevents frame loss due to congestion. Pressing the `Ctrl + S` keys enables the flow control and no output will be seen on the XR console until resumed.

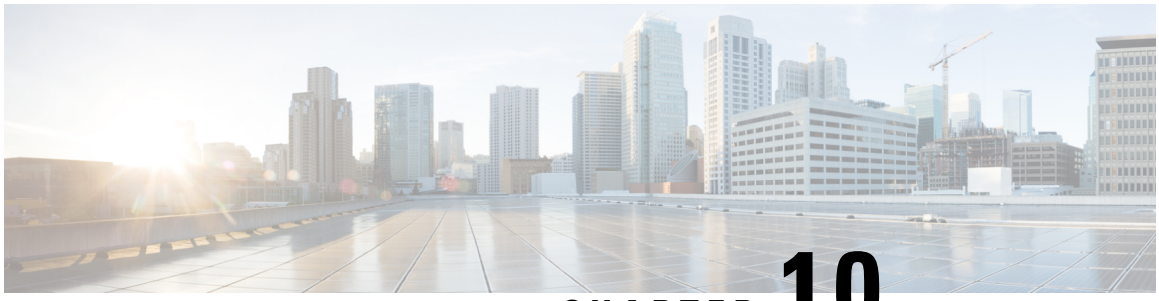
Solution:

Reset the console prompt.

[4] The **bash** and **run** commands can launch a `bash` shell, which the users can use to move files around the device on the active RP.

Procedure

Press the `Ctrl + Q` keys to resume the console output.



CHAPTER 10

Install Owner and Partner RPMs using IOS XR Install Infrastructure

This chapter describes how to install and manage Owner and Partner RPMs using the IOS XR install infrastructure.

This document is intended for network operators who want to install and manage proprietary Owner and Partner RPMs on devices running the Cisco IOS XR software. It assumes that these users are familiar with the basic concepts and commands of IOS XR and docker.

This chapter includes these sections:

- Owner and partner RPMs, on page 179
- Third-party unsigned RPMs, on page 187
- Troubleshooting Installation Failures, on page 189

Owner and partner RPMs

Owner RPMs are RPMs created by Cisco customers, or by other third parties. Cisco's customer (also known as the Owner) is responsible for the content and GPG-based signing of these RPMs as needed.

Partner RPMs are RPMs created by Cisco partners. These are supplied by Cisco and are signed with Cisco-managed security keys.

The owner and partner RPMs are installed as docker container images that run on the router.

The following steps provide the high level overview of the end-to-end signed owner RPM secure onboarding:

1. Owner signed RPM (docker image) in GISO or standalone needs owner's GPG keys to validate the RPM packages.
2. Owner's GPG keys are onboarded through secure workflow using key packages (explained in later sections).
3. Validation of the said key packages needs owner's public keys installed on the device.
4. Owner's public keys are onboarded on to the device using the ownership certificate (OC) or ownership voucher (OV) workflow as part of device ownership.
5. The chain of trust resides with Cisco through Ownership Vouchers.
6. The owner and partner RPMs are managed as docker container images using Cisco App Manager that are then run on the router. For more information, see the Application Hosting Configuration Guide.

Table 36: Feature History Table

Feature Name	Release Information	Feature Description
Signature verification for owner RPMs using owner public keys	Release 26.1.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q100, Q200, P100], 8700 [ASIC: P100, K100], 8010 [ASIC: A100]); Centralized Systems (8600 [ASIC:Q200]) ; Modular Systems (8800 [LC ASIC: Q100, Q200, P100]) This feature ensures integrity of owner RPMs and validates the authenticity of these RPMs by enabling the router to securely verify the owner RPM signatures using owner public keys onboarded on to the router. With this feature, you can now control whether or not to enable signature verification for owner RPMs based on defined security postures. When the signature verification for owner RPMs is enabled, all owner RPMs must be verified prior to IOS XR package installation. This feature introduces these changes: CLI: • A new command, **show install rpmdb keys**, has been added. • A new keyword, **keys**, has been added to the **show install active** and **show install committed** commands.

Feature Name	Release Information	Feature Description
Install Owner and Partner RPMs using IOS XR Install Infrastructure	Release 24.4.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM
Install Owner and Partner RPMs Using IOS XR Install Infrastructure	Release 24.2.11	You can now use the existing IOS XR install infrastructure to install your proprietary Owner and Partner RPMs. This enhancement streamlines the process of integrating third-party software seamlessly into the IOS XR environment, including bundling the owner and partner RPMs into a GISO. In previous releases, you could only install Owner and Partner applications using the Application Manager interface. This feature introduces the keyword **skip-implicit-owner-packages-checks** in the following install commands: • **install package** • **install replace** • **install source** • **install rollback** • **install replace reimage**

Signature verification for owner RPMs

Starting from Cisco IOS XR Release 26.1.1, you can choose whether to enable signature verification for owner RPMs or not. When the signature verification for owner RPMs is enabled, all owner RPMs must be signed with GPG keys and the signatures are always verified. Owner RPM signatures are verified using owner public

keys onboarded on to the router. Signature verification validates the authenticity and integrity of the RPMs, ensuring that they are correct and unmodified.

You can control whether or not owner RPM signatures are verified, by setting the security posture to Low, Medium, or High.

The supported security postures are:

- High: Owner RPMs must always be signed. The owner RPMs are installed only after the signatures are verified.
- Medium: Owner RPM signatures are not verified, but owner RPMs must be installed explicitly.

  In an explicit installation, owner RPMs can either be named (**install package add** *owner-foo*) or the **skip-implicit-owner-packages-checks** option be added (**install package add** <*iso*>**skip-implicit-owner-packages-checks**).

  This is the default security posture.
- Low: Owner RPMs signatures are not verified.

You can request the router to change from one security posture to another security posture. For more information, see Security posture transitions, on page 183 .

From Cisco IOS XR Release 26.1.1, a new command, **show install rpmdb keys**, has been introduced and a new keyword **keys**,has been added to the **show install active** and **show install committed** commands.

Limitations and Guidelines

General Limitations and Guidelines

- Owner and partner RPMs can be installed only on route processors, and not on the line cards.
- Owner and partner RPMs are managed by App Manager, which is responsible for running the docker containers and handling notifications from Install.
- Owner and partner RPMs must install files only to the designated filesystem locations. These locations are `/opt/owner/` and `/opt/partner/` respectively.
- When installing Owner and partner RPMs, the maximum size of a GISO that can be used for bootstrap or when using the **install replace** command is 4GB.
- If you are upgrading from an IOS XR release that does not support installing the Owner or Partner RPMs (for example, upgrading from IOS XR Release 24.1.1 to IOS XR Release 24.2.11), owner and partner RPMs will not be installed automatically during the upgrade process. See the *Two-Step Upgrade Process for Installing Owner or Partner RPMs* section for more information.

Limitations and Guidelines for Owner RPMs

- Names of all Owner RPMs must begin with the string “owner-”.
- Owner RPMs can either be unsigned or signed with a non-Cisco key. Owner RPM signatures can either be verified or not verified during the installation of these RPMs.
- Owner RPMs must not include any RPM scriptlets, including pre-install and post-install scripts.
- You must request the installation of an Owner RPMs either by listing them explicitly or by including an additional parameter *skip-implicit-owner-packages-checks* in the install commands.

- During the network boot process using PXE or when booting from a USB drive, Owner RPMs are not automatically installed. This limitation arises because in these scenarios, there is no mechanism for the owner (Cisco customer) to convey consent for the installation of Owner RPMs. That is, there is no way for the owner to provide the *skip-implicit-owner-packages-checks* parameter.

  If you want to include Owner RPMs after booting from an ISO via PXE or USB, execute the **install replace** command with the ISO that contains the desired Owner RPM packages. This action incorporates the Owner RPM packages into your installation without affecting any other aspects of the system.

- To onboard owner RPMs on a new RP, the router either requires a separate OV in the tar file or a separate MASA service to create an ownership voucher (OV), which can added to the router using XR interfaces.

Limitations and Guidelines for Partner RPMs

- All Partner RPMs must be signed with a Cisco key; otherwise, they are treated as owner RPMs.
- Names of all Partner RPMs must begin with the string “partner-”.

Security posture transitions

A security posture transition is the process of changing a system’s operational security mode for a feature from one predefined level to another. Security posture transitions ensure integrity and protection of the system throughout the change.

The router blocks the change from a Low or Medium security posture to a High security posture, if there are any unverifiable packages.

Security posture transitions are performed using ownership vouchers (OVs). For more informtion, see Security profiles for Cisco IOS XR software.

Installing owner and partner RPMs

This process provides the workflow to install the owner and partner RPMs.

Summary

Owner RPMs created by Cisco customers or by other third parties, and partner RPMs created by Cisco partners are installed as docker container images that run on the router.

Workflow

These are the stages of the owner and partner RPMs installation:

1. Create an RPM containing the application (in the form of a docker container image), according to the requirements for owner and partner RPMs.

Note Starting from Cisco IOS XR 26.1.1, prior to installing owner RPMs, owner RPMs must be signed and the owner RPM signatures must be verified if the security posture is set to High.

You can also include the RPMs in the GISO using the appropriate Cisco tools.

2. Install the RPMs using the XR Install infrastructure, that is, by using any of the install commands, similar to installing IOS XR RPMs.

Note When installing an owner RPM and when the security posture is set to Medium, you must do one of the following:

- Option 1: Specify the name of the owner RPM explicitly.

  ```
  install package add owner-foo
  ```

- Option 2: Specify the **skip-implicit-owner-packages-checks** option.

  ```
  install replace [iso] skip-implicit-owner-packages-checks
  ```

3. Activate the RPMs (or let the system do this automatically if requested in the install operation).
4. Commit the transaction (or let the system do this automatically).

Install unsigned owner RPMs

This task helps you to install unsigned owner RPMs using different command options.

Procedure

Use one of the following options to install owner and partner RPMs:

- Execute the **install package add** command to add one or more packages to the active software without replacing the entire software.

  Example 1: This example shows how to install, activate, and commit an owner RPM individually.

  ```
  Router# install package add disk0:owner-app-24.2.11.x86_64.rpm activate commit
  ```

  Example 2: This example includes the `skip-implicit-owner-packages-checks` option, which is required only if the owner RPM name is not specified explicitly. This example is applicable to owner RPMs for which the security posture is set to Medium.

  ```
  Router# install package add skip-implicit-owner-packages-checks activate commit
  ```

- Execute the **install replace** command to replace the currently installed software with a new GISO that includes new owner RPMs.

  ```
  Router# install replace /harddisk:/8000-64-24.2.11-owner.iso activate commit
  Router# install replace skip-implicit-owner-packages-checks activate commit
  ```

- Execute the **install rollback** command to roll back to roll back the installation of owner and partner RPMs in the software associated with the specific transaction ID.

  This example includes the `skip-implicit-owner-packages-checks` option, which is required only if the owner RPM name is not specified explicitly.

  ```
  Router# install rollback skip-implicit-owner-packages-checks
  ```

The owner RPMs are installed successfully.

In addition to the commands used in this section, the install infrastructure also supports the following install operations to install owner RPMs – **install package remove**,**install package deactivate**,and **install source**.

For information about using the XR application manager to control the life cycle of installed owner or partner RPMs, see the *Customize Docker Run Options Using Application Manager* section in the *Application Hosting Configuration Guide for Cisco 8000 Series Routers*.

Install signed owner RPMs with or without signature verification

Before you begin

- The signing keys must be installed. For more information, see Install key packages on the router. The owner keys or signing keys can also be included in a GISO.
- Security posture is set to High, Medium, or Low.

This task provides the steps to install signed owner RPMs with or without verifying their signatures.

Procedure

Step 1 Create owner RPMs as per the owner RPM requirements.

Step 2 Create a GPG key or owner key anchored to the ownership certificate (OC) and install the owner keys on the router.

Step 3 Sign the owner RPMs with the GPG keys created in the previous step and install these keys on the router using the key package. For installing key packages on the router, see Install key packages on the router.

Step 4 Install the owner RPMs using one of the installation commands used in Install unsigned owner RPMs. The router installs the signed owner RPMs based on the security postures:

- If the security posture is High, the router verifies the owner RPMs signatures using the signing keys installed on the router and installs the owner RPMs upon successful signature verification.
- If the security posture is Medium, the router installs the signed owner RPMs explicitly without verifying the owner RPM signatures.

  An explicit installation requires the owner RPM name specified in the installation command. The **skip-implicit-owner-packages-checks** option is required in an explicit installation, if the owner RPM name is not specified in the installation command.
- If the security posture is Low, the router installs the signed owner RPMs implicitly without verifying the owner RPM signatures.

  An implicit installation operation installs the owner RPMs in the GISO using the **install replace**<*iso*> command, where the packages to be installed are not listed explicitly.

Install signed owner RPMs using GISO

This task helps you to install signed owner RPMs using GISO.

Procedure

Step 1 Create a key package with GPG keys used to sign the owner RPM and sign it with a key anchored in the OC.

For more information about key packages, OC, and OV, see *Provision Third-party Key Packages* and *Device Ownership* chapters in the *System Security Configuration Guide*

Step 2 Create an owner RPM and sign it with the GPG key obtained using the key package.

Step 3 Include both the owner RPM and the signed key package in a GISO.

Step 4 Execute the **install replace** command to install the GISO image.

Example:

```
Router# install replace /harddisk:/ncs540l-giso-x64.iso
```

Step 5 Starting from Cisco IOS XR Release 26.1.1, the router verifies the signature on the key package. On successful verification, the router temporarily installs the GPG keys on the router.

Step 6 Starting from Cisco IOS XR Release 26.1.1, if the security posture is High, the router installs the RPMs from the GISO and verifies the signature for owner RPMs using the key installed earlier.
The router aborts the install operation if the signature verification for key packages or owner RPMs fails.

Step 7 Activate the installed software.

Step 8 Commit the changes.

Note
If the router is reloaded or the install operation is aborted before committing the changes, the current active key is discarded.

The router stores the installed key after committing the changes.

The signed key packages and owner RPMs are installed and verified successfully. The router updates the file system with the new software.

For information about using the XR application manager to control the life cycle of installed owner or partner RPMs, see the *Customize Docker Run Options Using Application Manager* section in the *Application Hosting Configuration Guide for Cisco 8000 Series Routers*.

Install signed owner RPMs using USB or iPXE boot

This task helps you to install owner RPMs using USB or iPXE boot.

Procedure

Step 1 Create a key package with GPG keys used to sign the owner RPM and sign it with a key anchored in the OC.

For more information about key packages, OC, and OV, see *Provision Third-party Key Packages* and *Device Ownership* chapters in the *System Security Configuration Guide*.

Step 2 Create an owner RPM and sign it with the GPG key obtained using the key package.

Step 3 Include the partner RPM, owner RPM, and the signed key package in a GISO.

Step 4 Boot the router using USB or iPXE.

Step 5 Starting from Cisco IOS XR Release 26.1.1, the bootstrap software from GISO verifies the signature on the key package. On successful verification, the bootstrap software temporarily installs the GPG keys on the router.

Step 6 Starting from Cisco IOS XR Release 26.1.1, if the security posture is High, the bootstrap software from GISO installs the RPMs from GISO and verifies the signature for owner RPMs using the key installed earlier.
Failure to onboard keys or install owner or partner RPMs does not block the system boot; the system continues to come up.

Step 7 The router reloads and boots from the disk.

The signed key packages and owner RPMs are installed and verified successfully. The router updates the file system with the new software.

For information about using the XR application manager to control the life cycle of installed owner or partner RPMs, see the *Customize Docker Run Options Using Application Manager* section in the *Application Hosting Configuration Guide for Cisco 8000 Series Routers*.

Two-step upgrade process for installing owner or partner RPMs

If you are upgrading from an IOS XR release that does not support installing the owner or partner RPMs (any release prior to IOS XR Release 24.2.11), these RPMs will not be installed automatically during the upgrade process.

This is because the installation process governed by the previous releases does not have the functionality to handle the new RPMs.

To install Owner or partner RPMs, you must perform the upgrade in two distinct steps:

1. Upgrade from the previous release to IOS XR Release 24.2.11 without the owner or partner RPMs.
2. Perform an additional upgrade using the same newer version, that is, IOS XR Release 24.2.11 to IOS XR Release 24.2.11. This time explicitly by including the owner or partner RPMs.

These upgrade steps can be accomplished using the same ISO that contains the new version (for example, IOS XR Release 24.2.11) and the additional Owner or Partner RPMs. This two-step process ensures that the new features are properly implemented and that the Owner or Partner RPMs are correctly installed on your system.

Starting from Cisco IOS XR Release 26.1.1, the initial upgrade is done without owner RPMs, partner RPMs, and their respective keys. In the subsequent upgrade to Cisco IOS XR Release 26.1.1 includes the signed owner RPMs, signed partner RPMs, and their respective keys. After the initial upgrade, the security posture defaults to Medium so that the unsigned owner RPMs are installed upon request. However, during the second upgrade where if the security posture is set to High, the router installs and verifies the owner RPMs implicitly. If the security posture is set to Low during the second upgrade, the router installs unsigned owner RPMs implicitly.

Third-party unsigned RPMs

An unsigned third-party RPM is a software package that

- originates from a source other than the official operating system vendor
- lacks a digital signature, and

- prevents verification of its authenticity and integrity.

Table 37: Feature History Table

Feature Name	Release Information	Feature Description
Third-party unsigned RPMs	Release 25.3.1	You can now install unsigned third-party RPMs on Cisco IOS XR systems, leveraging the existing install infrastructure and Golden ISO (GISO). This simplifies the deployment of custom or third-party containerized applications via the command-line interface, provided these unsigned RPMs adhere to Owner RPM conventions.

Guidelines for installing unsigned third-party RPMs

Use these guidelines when installing unsigned third-party RPMs.

- Third-party RPMs can be unsigned, and no signature verification is performed during installation for these RPMs.
- Do not include any RPM scriptlets, such as pre-install or post-install scripts.
- Third-party RPMs must install files only to `/opt/owner/` as per system conventions.
- Third-party unsigned RPMs are not installed implicitly during boot. To install third-party unsigned RPMs, use the **install replace** command with the **skip-implicit-owner-packages-checks** keyword.
- During PXE or USB boot, third-party owner RPMs are not installed automatically. Install them after initial boot using an explicit install command.
- The maximum supported GISO size, including third-party owner RPMs, is 4 GB.

Install third-party unsigned RPMs

Before you begin

- Ensure the RPM follows the owner RPM requirements:
  - The name starts with `owner-`.
  - It installs only to allowed locations.
  - It contains no scriptlets.
- The RPM file must be available in a GISO, which can reside in `disk0` or harddisk or an external repository.

Procedure

Step 1 Install all third-party unsigned RPMs included in the GISO using the **skip-implicit-owner-packages-checks** command.

Example:

```
Router# install replace /harddisk:/ncs5500-custom-25.3.1.iso skip-implicit-owner-packages-checks
commit
```

Step 2 Verify installation of third-party unsigned RPMs.

Example:

```
Router# show install active summary
Label : 25.3.1.06I-V1_TPA

    Active Packages: 15
        ncs5500-xr-25.3.1.06I version=25.3.1.06I [Boot image]
        ncs5500-isis-1.0.0.0-r253106I
        ncs5500-eigrp-1.0.0.0-r253106I
        ncs5500-li-1.0.0.0-r253106I
        ncs5500-xcare-1.0.0.0-r253106I
        ncs5500-mpls-te-rsvp-1.0.0.0-r253106I
        ncs5500-ospf-1.0.0.0-r253106I
        ncs5500-healthcheck-1.0.0.0-r253106I
        ncs5500-mgbl-1.0.0.0-r253106I
        ncs5500-mcast-1.0.0.0-r253106I
        ncs5500-lictrl-1.0.0.0-r253106I
        ncs5500-k9sec-1.0.0.0-r253106I
        ncs5500-mpls-1.0.0.0-r253106I
        owner-swan-0.1.3-25.3.1.x86_64 [TPA]
        owner-bonnet-0.1.2-25.3.1.x86_64 [TPA]
```

Troubleshooting Installation Failures

Normal Installation Failures (While IOS XR is Running)

If you encounter a failure while installing Owner or Partner RPMs during routine operations, such as **install replace** or **install package add**, the system treats this failure exactly like the failure to install any IOS XR RPMs.

The installation is aborted, and you must resolve the issue to continue. Standard recovery procedures for failed RPM installations should be followed in this case. For more information, see the *Troubleshoot Router Setup and Upgrade* chapter.

Bootstrap Installation Failures (During System Reimage)

If an Owner or Partner RPM fails to install during the bootstrap process, such as during a reimage from a disk, the system continues to boot, assuming all IOS XR software have been installed successfully.

After the system restarts, you will be alerted if any Owner or Partner RPMs were not installed. Use the **show install boot packages-not-installed** command for a list of uninstalled packages and the reasons they were not applied.

This distinction exists because the IOS XR system can fully boot and operate without the Owner or Partner RPMs, allowing you to troubleshoot the issue in a fully operational environment.

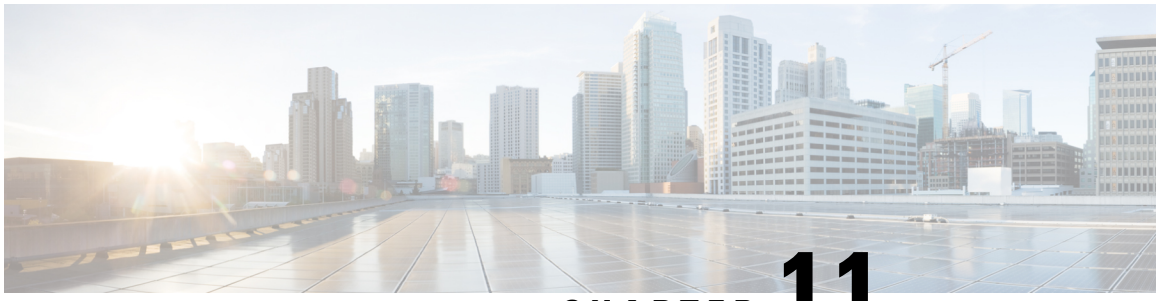
Inconsistencies in Owner/Partner RPMs Across Route Processors

If the system detects inconsistent versions of the IOS XR software on the active and standby RPs, the standby RP is repeatedly reimaged until the versions match.

However, if the system detects different versions of Owner or Partner applications on the active and standby route processors (RPs), you are required to address the discrepancy.

These are some of the use cases:

1. Use case 1:If the standby RP has Owner or Partner RPM versions different from that of an active RP, the standby RP is reimaged.
2. Use case 2: If both the standby RP and active RP have same Owner or Partner RPM versions, no further action is required.
3. Use case 3: If both the standy RP and active RP have different Owner or Partner RPM versions, syslogs are generated to notify users about different RPM versions.



CHAPTER 11

Upgrading Field-Programmable Device

An FPD is a field programmable logic device which contains non-volatile, re-programmable memory to define its internal wiring and functionality. The contents of this non-volatile memory are called the FPD image or FPD firmware. Over the lifespan of an FPD, FPD firmware images may need upgrades for bug fixes or functionality improvements. These upgrades are performed in the field with minimum system impact.

- Overview of FPD Image Upgrade , on page 191
- Restrictions for FPD Upgrade , on page 191
- Types of FPD Upgrade Service, on page 192
- How to Upgrade FPD Images, on page 194
- Automatic Line Card Reload on FPD Upgrade, on page 200
- Power Module Upgrades, on page 200
- Upgrading FPD for PSU, on page 202

Overview of FPD Image Upgrade

An FPD image is used to upgrade the software on an FPD.

Every new IOS XR software package includes FPD images. These FPD images are automatically upgraded when you upgrade the Cisco IOS XR software image. You can also upgrade FPD images manually when required.

FPD versions must be compatible with the Cisco IOS XR software that is running on the router; if an FPD version is incompatible with the Cisco IOS XR software, the device with the FPGA may not operate properly.

Restrictions for FPD Upgrade

The Optics FPD Upgrade Service is not available using the **upgrade hw-module fpd** command.

You can upgrade Optics FPD using the **upgrade optics port filename /harddisk:/cl1.bin location** command.

For more information on optics FPD upgrade, see *Upgrade QDD Optical Modules* in Upgrade the Router Chapter in Cisco IOS XR Setup and Upgrade Guide for Cisco 8000 Series Routers.

Restrictions For Automatic FPD Upgrade

The following FPDs do not support Auto FPD Upgrade:

- Optics FPDs
- Power Module FPDs
- Timing FPDs

Types of FPD Upgrade Service

An FPD image package is used to upgrade FPD images. The **install activate** command is used to place the FPD binary files into the expected location on the boot devices.

Supported Upgrade Methods

Method	Remarks
Manual Upgrade	Upgrade using CLI, force upgrade supported.
Auto Upgrade	Upgrade using install SMU activation or during image upgrade. User can enable/disable auto upgrade feature.

Manual FPD upgrade

Manual FPD upgrade is performed using the **upgrade hw-module fpd** command. All cards or all FPGA in a card can be upgraded. If reload is required to activate FPD, the upgrade should be complete. Line-cards, fabric cards and RP cardsInterface module (IMs) and RSPs can't be reloaded during the process of the FPD upgrade.

FPD upgrade is transaction-based:

- Each fpd upgrade CLI execution is one transaction.
- Only one transaction is allowed at any given time.
- One transaction may include one or many FPD upgrades.

Once the upgrade is complete, the router/the card (on which the FPD is upgraded) must be reloaded.

The **force** option can be used to forcibly upgrade the FPD (regardless of whether it's required or not). It triggers all FPDs to be upgraded or downgraded. The **force** option can also be used to downgrade or upgrade the FPGAs even after the version check. However, the **force** option must be used cautiously and only to recover a component from a failed upgrade.

Note

- Sometimes, FPDs can have primary and backup images.
- The use of the **force** option when performing an FPD upgrade isn't recommended except under explicit direction from Cisco engineering or TAC for a one-time purpose only.
- A new FPD upgrade should be issued only when previous FPD upgrades have been completed on the same FPD with the following syslog message:

```
RP/0/RP0/CPU0:May 10 10:11:44.414 UTC: fpd-serv[205]: %INFRA-FPD_Manager-1-UPGRADE_ALERT
 : FPD Upgrade Completed (use "show hw-module fpd" to check upgrade status)
```

Automatic FPD Upgrade

FPD auto-upgrade is enabled by default. To ensure that the FPD image is automatically upgraded, you should not disable this feature. If you need to disable the automatic upgrade of the FPD image running on the Field Replaceable Unit (FRU), you can manually apply the configuration **fpd auto-upgrade disable** in administration configuration mode.

With FPD auto-upgrade enabled, FPD images are automatically updated in the following instances:

- Software upgrade is carried out.
- Field Replaceable Unit(FRU) such as Line cards, RSPs, Fan Trays or alarm cards are added to an existing router or reloaded.

For the automatic FPD upgrade to work on a system upgrade, the following conditions must be met:

- The FPD package installation envelope (PIE) must be installed on the router.
- The FPD PIE must be activated together with the new Cisco IOS XR image.

For the automatic FPD upgrade to work on a FRU Insertion or reload , the following conditions must be met:

- The FPD package installation envelope (PIE) must be installed and activated on the router.



Note Although the FPD upgrade is performed during the install operation, there is no install commit performed. Therefore, once the FPD has been upgraded, if the image is rolled back to the original version, the FPD version is not downgraded to the previous version.

The automatic FPD upgrade is not performed in the following instances:

- Line cards or other cards or alarm cards are added to an existing router.
- A line card chassis is added to an existing router.
- A non-reload software maintenance upgrade (SMU) or PIE installation is performed, even where the FPD image version changes. Since a non-reload installation is, by definition, not supposed to reload the router, and an FPD upgrade requires a router reload, the automatic FPD upgrade is repressed.



Note In all cases where the automatic FPD upgrade is not performed, you must perform a manual FPD upgrade using the **upgrade hw-module fpd** command.

FPD auto-upgrade can be enabled and disabled. When auto FPD is enabled, it automatically updates FPDs when a SMU or image changes, including an updated firmware revision. Use the **fpd auto-upgrade** command to disable or enable auto-fpd.

YANG Data Models for Auto FPD Upgrade

YANG is a data modeling language that helps to create configurations, retrieve operational data and execute actions. The router acts on the data definition when these operations are requested using NETCONF RPCs. The data model handles the following types of requirements on the routers for FPD:

Operational Data	Native Data Model	CLI Commands
Auto Upgrade: Enabling or disabling of automatic upgrade of FPD.	Cisco-IOS-XR-fpd-infra-cfg.yang	• **fpd auto-upgrade enable** • **fpd auto-upgrade disable**
Auto Reload: Enabling or disabling of automatic reload of FPD.	Cisco-IOS-XR-fpd-infra-cfg.yang	• **fpd auto-reload enable** • **fpd auto-reload disable**

You can access the data models from the Github repository. To learn more about the data models and put them to use, see the *Programmability Configuration Guide for Cisco 8000 Series Routers*.

How to Upgrade FPD Images

The main tasks of the FPD upgrade service are:

- Check FPD image version to decide if a specific firmware image needs an upgrade or not.

  You can determine if an FPD image upgrade is needed using the **show hw-module fpd** command and perform the upgrade, if needed, under the following circumstances:

  - Migrate the software to a later Cisco IOS XR software release.
  - Swap line cards from a system running a different Cisco IOS XR software release.
  - Insert a new line card.

- Automatic FPD Image Upgrade (if enabled) Or Manual FPD Image Upgrade using the **upgrade hw-module fpd** command.
- Invoke the appropriate device driver with a name of the new image to load.

Guidelines for Upgrading FPD

The following are some of the important guidelines to be taken into consideration for upgrading FPD:

- Upgrades to the Cisco IOS XR software might result in an FPD incompatibility. Ensure that you perform the FPD upgrade procedure and resolve all incompatibilities, for the cards to function properly.
- The use of the **force** option when performing a FPD upgrade is not recommended except under explicit direction from Cisco engineering or TAC for a one-time purpose only.
- If your card supports multiple FPD images, you can use the **show fpd package** admin command to determine what specific image to upgrade in the **upgrade hw-module fpd** command.
- A message is displayed when router modules cannot get upgraded during upgrade with **location all** option indicating that the FPGA is intentionally skipped during upgrade. To upgrade such FPGAs, you can use the CLI command with a particular location explicitly specified. For example, **upgrade hw-module fpd all location 0/3/1**.
- 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.

Before you begin

- Before performing the manual upgrading the FPD on your router using the **upgrade hw-module FPD** , you must install and activate the fpd.pie and fpd.rpm package.
- The FPD upgrade procedure is performed while the card is online. At the end of the procedure the card must be reloaded before the FPD upgrade is complete. To reload the card, you can use the **hw-module location** *location* **reload** command in Config mode, during the next maintenance window. The upgrade procedure is not complete until the card is reloaded.
- During the FPD upgrade, you *must not* do the following:
  - Reload, perform an online insertion and removal (OIR) of a line card (LC), or power down the chassis. Doing so may cause the node to enter an unusable state.
  - Press **Ctrl-C** if the console appears to hang without any output. Doing so may abort the upgrade.
- If you are not sure whether a card requires an FPD upgrade, you can install the card and use the **show hw-module fpd** command to determine if the FPD image on the card is compatible with the currently running Cisco IOS XR software release.

Procedure

Step 1 **show hw-module fpd location** {**all** | *node-id*}

Example:

```
Router#show hw-module fpd location all
```

or

```
Router#show hw-module fpd location 0/4/cpu0
```

Displays the current FPD image versions for the specified card or all cards installed in the router. Use this command to determine if you must upgrade the FPD image on your card.

In the event of an FPD incompatibility with your card, you might receive the following error message:

```
LC/0/0/CPU0:Jul 5 03:00:18.929 UTC: optics_driver[220]: %L2-OPTICS-3-BAD_FPGA_IMAGE : Detected bad
MI FPGA image programmed in MI FPGA SPI flash in 0/0/CPU0 location: Failed to validate meta data CRC

LC/0/0/CPU0:Jul 5 03:00:19.019 UTC: optics_driver[220]: %L2-OPTICS-3-BACKUP_FPGA_LOADED : Detected
Backup FPGA image running on 0/0/CPU0 - primary image corrupted (@0x8c = 0x44)
RRouter:Jul 5 03:00:48.987 UTC: fpd-serv[301]: %PKT_INFRA-FM-3-FAULT_MAJOR : ALARM_MAJOR
:FPD-NEED-UPGRADE :DECLARE :0/0:
```

Step 2 (Optional) **show fpd package**

Example:

The following example shows a sample output from the **show fpd package** command:

```
Router#show fpd package
================================ ============================================
                                        Field Programmable Device Package
                                 ============================================
                                           Req     SW      Min Req   Min Req
Card Type            FPD Description       Reload  Ver     SW Ver  Board Ver
```

```
================== ========================== ====== ======= ======== =========
------------------------------------------------------------------------------
8201               Bios                       YES     1.23      1.23      0.0
                   BiosGolden                 YES     1.23      1.15      0.0
                   IoFpga                     YES     1.11      1.11      0.1
                   IoFpgaGolden               YES     1.11      0.48      0.1
                   SsdIntelS3520              YES     1.21      1.21      0.0
                   SsdIntelS4510              YES    11.32     11.32      0.0
                   SsdMicron5100              YES     7.01      7.01      0.0
                   SsdMicron5300              YES     0.01      0.01      0.0
                   x86Fpga                    YES     1.05      1.05      0.0
                   x86FpgaGolden              YES     1.05      0.48      0.0
                   x86TamFw                   YES     5.13      5.13      0.0
                   x86TamFwGolden             YES     5.13      5.05      0.0
------------------------------------------------------------------------------
8201-ON            Bios                       YES     1.208     1.208     0.0
                   BiosGolden                 YES     1.208     1.207     0.0
                   IoFpga                     YES     1.11      1.11      0.1
                   IoFpgaGolden               YES     1.11      0.48      0.1
                   SsdIntelS3520              YES     1.21      1.21      0.0
                   SsdIntelS4510              YES    11.32     11.32      0.0
                   SsdMicron5100              YES     7.01      7.01      0.0
                   SsdMicron5300              YES     0.01      0.01      0.0
                   x86Fpga                    YES     1.05      1.05      0.0
                   x86FpgaGolden              YES     1.05      0.48      0.0
                   x86TamFw                   YES     5.13      5.13      0.0
                   x86TamFwGolden             YES     5.13      5.05      0.0
------------------------------------------------------------------------------
8201-SYS           Bios                       YES     1.23      1.23      0.0
                   BiosGolden                 YES     1.23      1.15      0.0
```

Displays which cards are supported with your current Cisco IOS XR software release, which FPD image you need for each card, and what the minimum hardware requirements are for the various modules. (A minimum hardware requirement version of 0.0 indicates that all hardware can support this FPD image version.)

If there are multiple FPD images for your card, use this command to determine which FPD image to use if you want to upgrade only a specific FPD type.

The FPD name used in the FPD Description column of the output of the **show fpd package** command includes the last ten characters of DCO-PID. Depending on the slot and port numbers, the FPD name is appended with DCO_0, DCO_1, or DCO_2. For example, the FPD names for `CFP2-WDM-D-1HL` in port 0 and port 1 are `-WDM-D-1HL_DCO_0` and `WDM-D-1HL_DCO_1` respectively.

Step 3 **upgrade hw-module fpd** {**all** | *fpga-type*} [**force**] **location** [**all** | *node-id*]

Example:

```
Router#upgrade hw-module fpd
all location 0/3/1
.
.
.
Successfully upgraded 1 FPD for SPA-2XOC48POS/RPR
  on location 0/3/1
```

```
Router#upgrade hw-module location 0/RP0/CPU0 fpd all
upgrade command issued (use "show hw-module fpd" to check upgrade status)
Router: %SECURITY-SSHD_SYSLOG_PRX-6-INFO_GENERAL : sshd[29745]: Accepted authentication for cisco
from 223.255.254.249 port 39510 ssh2
upgrade hw-module location 0/RP0/CPU0 fpd all RRouter: ssh_syslog_proxy[1223]:
%SECURITY-SSHD_SYSLOG_PRX-6-INFO_GENERAL : sshd[29803]: Accepted authentication for cisco from
223.255.254.249 port 39524 ssh2
```

```
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT : Upgrade for the following FPDs has been
committed:
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT : Location           FPD name
Force
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT :
================================================
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT : 0/RP0/CPU0         x86FpgaGolden
FALSE
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT : 0/RP0/CPU0         x86Fpga
FALSE
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT : 0/RP0/CPU0         SsdMicron5300
FALSE
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT : 0/RP0/CPU0         IoFpgaGolden
FALSE
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT : 0/RP0/CPU0         IoFpga
FALSE
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT : 0/RP0/CPU0         DbIoFpgaGolden
FALSE
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT : 0/RP0/CPU0         DbIoFpga
FALSE
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT : 0/RP0/CPU0         BiosGolden
FALSE
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT : 0/RP0/CPU0         Bios
FALSE
Router:fpd_client[385]: %PLATFORM-FPD_CLIENT-1-UPGRADE_SKIPPED : FPD upgrade skipped for
x86FpgaGolden@0/RP0/CPU0: Image not upgradable
Router:fpd_client[385]: %PLATFORM-FPD_CLIENT-1-UPGRADE_SKIPPED : FPD upgrade skipped for
x86TamFwGolden@0/RP0/CPU0: Image not upgradable
Router:fpd_client[385]: %PLATFORM-FPD_CLIENT-1-UPGRADE_SKIPPED : FPD upgrade skipped for
x86FpgaGolden@0/RP0/CPU0: A dependent FPD upgrade is skipped
Router:fpd_client[385]: %PLATFORM-FPD_CLIENT-1-UPGRADE_SKIPPED : FPD upgrade skipped for
IoFpgaGolden@0/RP0/CPU0: Upgrade not required
Router:fpd_client[385]: %PLATFORM-FPD_CLIENT-1-UPGRADE_SKIPPED : FPD upgrade skipped for
DbIoFpgaGolden@0/RP0/CPU0: Upgrade not required
Router:fpd_client[385]: %PLATFORM-FPD_CLIENT-1-UPGRADE_SKIPPED : FPD upgrade skipped for
BiosGolden@0/RP0/CPU0: Image not upgradable
Router:fpd_client[385]: %PLATFORM-FPD_CLIENT-1-UPGRADE_SKIPPED : FPD upgrade skipped for
SsdMicron5300@0/RP0/CPU0: Upgrade not required as it is current

Router#fpd_client[385]: %PLATFORM-FPD_CLIENT-1-UPGRADE_COMPLETE : FPD upgrade complete for
Bios@0/RP0/CPU0 [image upgraded to version 254.00]
Router:fpd_client[385]: %PLATFORM-FPD_CLIENT-1-UPGRADE_COMPLETE : FPD upgrade complete for
x86TamFw@0/RP0/CPU0 [image upgraded to version 7.10]
Router:fpd_client[385]: %PLATFORM-FPD_CLIENT-1-UPGRADE_COMPLETE : FPD upgrade complete for
DbIoFpga@0/RP0/CPU0 [image upgraded to version 14.00]
Router:fpd_client[385]: %PLATFORM-FPD_CLIENT-1-UPGRADE_COMPLETE : FPD upgrade complete for
IoFpga@0/RP0/CPU0 [image upgraded to version 14.00]
Router:fpd_client[385]: %PLATFORM-FPD_CLIENT-1-UPGRADE_COMPLETE : FPD upgrade complete for
x86Fpga@0/RP0/CPU0 [image upgraded to version 254.00]
Router:shelfmgr[459]: %PLATFORM-SHELFMGR-6-INFO_LOG : 0/RP0/CPU0 is operational
Router:fpd-serv[265]: %INFRA-FPD_Manager-1-UPGRADE_ALERT :  FPD Upgrade Completed (use "show hw-module
 fpd" to check upgrade status)
```

Upgrades all the current FPD images that must be upgraded on the specified card with new images.

Before continuing to the next step, wait for confirmation that the FPD upgrade has successfully completed. Status messages, similar to these, are displayed to the screen until the FPD upgrade is completed:

```
FPD upgrade started.
FPD upgrade in progress..
FPD upgrade in progress..
FPD upgrade sent to location xxxx
FPD upgrade sent to location yyyy
```

```
FPD upgrade in progress..
FPD upgrade finished for location xxx
FPD upgrade in progress..
FPD upgrade finished for location yyyy
FPD upgrade completed.
```

The “FPD upgrade in progress.” message is printed every minute. These logs are information logs, and as such, are displayed if the **logging console informational** command is configured.

If Ctrl-C is pressed while the FPD upgrade is in progress, the following warning message is displayed:

```
FPD upgrade in progress on some hardware,
aborting now is not recommended as it might
cause HW programming failure and result in
RMA of the hardware.
Do you want to continue? [Confirm(y/n)]
```

If you confirm that you want to abort the FPD upgrade procedure, this message is displayed:

```
FPD upgrade process has been aborted, please
check the status of the hardware and reissue
the upgrade command if required.
```

Note

- If your card supports multiple FPD images, you can use the **show fpd package** admin command to determine what specific image to upgrade in the **upgrade hw-module fpd** command.
- A message is displayed when router modules cannot get upgraded during upgrade with **location all** option indicating that the FPGA is intentionally skipped during upgrade. To upgrade such FPGAs, you can use the CLI command with a particular location explicitly specified. For example, **upgrade hw-module fpd all location 0/3/1**.
- 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.

Step 4 **hw-module location** { *node-id* | **all** } **reload**

Use the **hw-module location reload** command to reload a line card.

```
Router:ios(config)# hw-module location 0/3 reload
```

Step 5 **exit**

Step 6 **show hw-module fpd**

Verifies that the FPD image on the card has been successfully upgraded by displaying the status of all FPDs in the system.

Example:

```
Router# show hw-module fpd
Auto-upgrade:Disabled
Attribute codes: B golden, P protect, S secure, A Anti Theft aware
                                                                          FPD Versions
                                                                         ==============
Location   Card type             HWver FPD device       ATR Status   Running Programd  Reload Loc
-----------------------------------------------------------------------------------------------
0/RP0/CPU0 8201                  0.30  Bios                 NEED UPGD  7.01    7.01    0/RP0/CPU0
0/RP0/CPU0 8201                  0.30  BiosGolden       B   NEED UPGD          7.01    0/RP0/CPU0
```

```
0/RP0/CPU0 8201                0.30  IoFpga               NEED UPGD  7.01    7.01         0/RP0
0/RP0/CPU0 8201                0.30  IoFpgaGolden     B   NEED UPGD          7.01         0/RP0
0/RP0/CPU0 8201                0.30  SsdIntelS3520        NEED UPGD  7.01    7.01         0/RP0
0/RP0/CPU0 8201                0.30  x86Fpga              NEED UPGD  7.01    7.01         0/RP0
0/RP0/CPU0 8201                0.30  x86FpgaGolden    B   NEED UPGD          7.01         0/RP0
0/RP0/CPU0 8201                0.30  x86TamFw             NEED UPGD  7.01    7.01         0/RP0
0/RP0/CPU0 8201                0.30  x86TamFwGolden   B   NEED UPGD          7.01         0/RP0
0/PM0      PSU2KW-ACPI         0.0   PO-PrimMCU           NEED UPGD  7.01    7.01       NOT REQ
0/PM1      PSU2KW-ACPI         0.0   PO-PrimMCU           NEED UPGD  7.01    7.01       NOT REQ
```

If the cards in the system do not meet the minimum requirements, the output contains a “NOTES” section that states how to upgrade the FPD image.

Table 38: show hw-module fpd Field Descriptions

Field	Description
Card Type	Module part number.
HW Version	Hardware model version for the module.
Type	Hardware type. • lc—Line card
Subtype	FPD type. Can be one of the following types: • Bios - Basic Input/Output System • BiosGolden - Golden BIOS image • IoFpga - Input/Output Field-Programmable Gate Array • IoFpgaGolden - Golden IoFpga • SsdIntelS3520 - Solid State Drive, made by Intel, of the model series S3520 • x86Fpga - Field-Programmable Gate Array designed to work with x86-based systems • x86FpgaGolden - Golden image of x86Fpga • x86TamFw - x86 Tam firmware • x86TamFwGolden - Golden image of x86TamFw • PO-PrimMCU - Primary microcontroller unit associated with a 'PO'
Inst	FPD instance. The FPD instance uniquely identifies an FPD and is used by the FPD process to register an FPD.
Current SW Version	Currently running FPD image version.
Upg/Dng?	Specifies whether an FPD upgrade or downgrade is required. A downgrade is required in rare cases when the version of the FPD image has a higher major revision than the version of the FPD image in the current Cisco IOS XR software package.

Automatic Line Card Reload on FPD Upgrade

This feature automatically reloads a newly inserted line card (LC) after a successful FPD upgrade. The prior auto FPD upgrade process did not reload the line card automatically, the user had to manually reload the LC.

Restrictions for Automatic Line Card Reload on FPD Upgrade

The following restriction must be considered while configuring automatic line card reload on FPD upgrade:

- If the FPD upgrade fails on a line card then the automatic line card reload feature (if enabled) stops the LC from reloading.

Configure Automatic Line Card Reload on FPD Upgrade

The following sample shows how to configure auto-reload feature:

```
Router# config
Router(config)#fpd auto-upgrade enable
Router(config)#fpd auto-reload enable
Router(config)#commit
```

The auto-reload feature is only supported on line cards.

Note During the FPD upgrade process, the linecard may display IOS XR RUN state before triggering auto-reload.

Power Module Upgrades

In Cisco IOS XR Routers, Field Programmable Device (FPD) upgrades for power modules are used to update the firmware or hardware logic of power entry modules (PEMs) within the router. These upgrades ensure that power modules operate effectively with the latest enhancements and bug fixes. Follow the Manual Power Module FPD Upgrade procedure to upgrade the FPD on PEMs.

Manual Power Module FPD Upgrade

Manual Power modules FPD upgrades are supported on Cisco Routers and should be performed in Config mode only. This feature lets you perform FPD upgrades on individual PEMs.

Only power modules that support FPD upgrades can be upgraded manually.

Note Power module upgrades are time consuming and can't be implicitly upgraded or as a part of automatic FPD upgrades. These modules must be upgraded independent of the other fpga upgrades.

Note Make sure that all power feeds to the PSU are active for a successful FPD upgrade of power modules.

To determine which PEMs requires upgrade, use **show hw-module location all fpd**.

PEMs requiring upgrade are in **UPGD SKIP** status.

```
Router#show hw-module location all fpd

Auto-upgrade:Disabled
Attribute codes: B golden, P protect, S secure, A Anti Theft aware
                                                                                FPD Versions
                                                                               ==============
Location   Card type             HWver FPD device       ATR Status   Running Programd
Reload Loc
-----------------------------------------------------------------------------------------------
0/RP0/CPU0 8201                  0.30  Bios                 NEED UPGD  7.01    7.01
0/RP0/CPU0
0/RP0/CPU0 8201                  0.30  BiosGolden       B   NEED UPGD          7.01
0/RP0/CPU0
0/RP0/CPU0 8201                  0.30  IoFpga               NEED UPGD  7.01    7.01
  0/RP0
0/RP0/CPU0 8201                  0.30  IoFpgaGolden     B   NEED UPGD          7.01
  0/RP0
0/RP0/CPU0 8201                  0.30  SsdIntelS3520        NEED UPGD  7.01    7.01
  0/RP0
0/RP0/CPU0 8201                  0.30  x86Fpga              NEED UPGD  7.01    7.01
  0/RP0
0/RP0/CPU0 8201                  0.30  x86FpgaGolden    B   NEED UPGD          7.01
  0/RP0
0/RP0/CPU0 8201                  0.30  x86TamFw             NEED UPGD  7.01    7.01
  0/RP0
0/RP0/CPU0 8201                  0.30  x86TamFwGolden   B   NEED UPGD          7.01
  0/RP0
0/PM0      PSU2KW-ACPI           0.0   PO-PrimMCU           NEED UPGD  7.01    7.01
NOT REQ
0/PM1      PSU2KW-ACPI           0.0   PO-PrimMCU           NEED UPGD  7.01    7.01
NOT REQ
```

To upgrade the power modules manually, use [**admin**] **upgrade hw-module location 0/PT***location* **fpd** <fpd_device>.

```
Router# admin
Router(admin)# upgrade hw-module location 0/PT0 fpd PM0-DT-Pri0MCU
```

To force a power module upgrade, use **upgrade hw-module fpd all force location pm-all** command in **Admin** mode.

Upgrading FPD for PSU

Table 39: Feature History Table

Feature Name	Release Information	Feature Description
Optimized PSU FPD Upgrade	Release 7.8.1	We have optimized the upgrade process of Field-Programmable Devices (FPDs) associated with the Power Supply Unit (PSUs) on the router. During the installation and PSU insertion process on the router, the FPDs associated with the PSUs are automatically upgraded. Starting this release, the PSU FPDs are grouped in the form of a parent FPD and its related child FPDs, and the upgrade image is downloaded only once. The upgrade is then triggered on the parent FPD PSU and replicated to the child FPD PSUs. In earlier releases, you downloaded the FPD image for each FPD associated with that PSU, and the upgrade process was then triggered sequentially. This process was time-consuming. The feature is supported on the following PSUs: • PSU2KW-ACPI • PSU2KW-HVPI • PSU3KW-HVPI • PSU4.8KW-DC100

Note You must disable **auto FPD upgrade** for PSUs before upgrading the router to Cisco IOS XR Software Release 7.9.1 or later if your router uses any of the following PSUs:

- PSU2KW-ACPI
- PSU2KW-ACPE
- PSU2KW-HVPI
- PSU4.8KW-DC100

To disable auto FPD upgrade, use the following command:

fpd auto-upgrade exclude pm

```
RP/0/RSP0/CPU0:ios# show running-config fpd auto-upgrade
RP/0/RP0/CPU0:ios(config)#fpd auto-upgrade exclude pm
RP/0/RP0/CPU0:ios(config)#commit
RP/0/RP0/CPU0:ios#
```

Automatic FPD Upgrade for PSU

Feature Name	Release Information	Feature Description
Automatic FPD upgrade for PSU	Release 7.5.2	Automatic FPD upgrade for PSUs is now enabled. In earlier releases, automatic upgrades did not apply to FPDs associated with the PSUs.

During the Power Supply Unit (PSU) insertion and installation process, the routers can now automatically upgrade the Field-Programmable Devices (FPD) associated with the PSUs.

Starting with Cisco IOS-XR Release 7.5.2, the automatic FPD upgrade includes the FPDs associated with the PSUs by default. This means that when automatic FPD upgrade is enabled, the FPDs associated with the PSUs will also be upgraded. The upgrades for the PSUs will occur sequentially, so the FPD upgrades for the PSUs will take longer than for other components.

You can choose to exclude PSUs from the automatic upgrade process to reduce the time taken for FPD automatic upgrade by preventing them from being upgraded upon insertion or during a system upgrade using the **fpd auto-upgrade exclude pm** command.

Configuration example for excluding PSUs from automatic FPD upgrade:

Configuration

```
Router# config
Router(config)# fpd auto-upgrade enable
Router(config)# fpd auto-upgrade exclude pm
Router(config)# commit
```

Show Running Configuration

```
Router# show running-config fpd auto-upgrade
fpd auto-upgrade enable
fpd auto-upgrade include pm
```

Exclude the Default PSU Upgrade from the Automatic FPD Upgrade

Table 40: Feature History Table

Feature Name	Release Information	Feature Description
Exclude the Default PSU Upgrade from the Automatic FPD Upgrade	Release 24.3.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100])(select variants only*); Modular Systems (8800 [LC ASIC: P100])(select variants only*) To make the automatic FPD upgrade process more time efficient, we have decreased the default time needed for FPD automatic upgrades by excluding PSUs from the automatic upgrade process. This is because the PSU upgrades are carried out one after the other, and on a fully loaded router, the process could take more than an hour to complete. We have also added an option to include the PSU in the automatic FPD upgrade. Previously, the PSU upgrade was included by default in the automatic FPD upgrade. The feature introduces the following change: **CLI:** • The **include pm** keyword is introduced in the **fpd auto-upgrade** command. *This feature is supported on: • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM

The routers automatically upgrade the Field-Programmable Devices (FPDs) associated with the Power Supply Unit (PSU) by default during the PSU insertion and installation process.

Starting with Cisco IOS-XR Release 24.3.1, the automatic FPD upgrade excludes the FPDs associated with the PSUs by default. This means that when the automatic FPD upgrade is enabled, the FPDs associated with the PSUs will not be upgraded by default to avoid the FPD automatic upgrade taking longer. The PSU upgrade exclusion is because the PSU upgrades will occur sequentially, and the FPD upgrades for the PSUs will take longer for a fully loaded router.

You can include the PSU upgrade to the FPD automatic upgrade process using the **fpd auto-upgrade include pm** command.

Include PSUs to Automatic FPD Upgrade

To include the PSU upgrade to FPD automatic upgrade process, do the following:

Procedure

Step 1 Enable the FPD automatic upgrade.

Example:

```
Router# config
Router(config)# fpd auto-upgrade enable
Router(config)# commit
```

Step 2 Include PSU upgrade in the FPD automatic upgrade.

Example:

```
Router# config
Router(config)# fpd auto-upgrade include pm
Router(config)# commit
```

Step 3 Verify the FPD and PSU automatic upgrade configurations.

Example:

```
Router# show running-config fpd auto-upgrade
fpd auto-upgrade enable
fpd auto-upgrade include pm
```

Step 4 View the status of PSU auto upgrade.

Example:

```
Router# show hw-module fpd

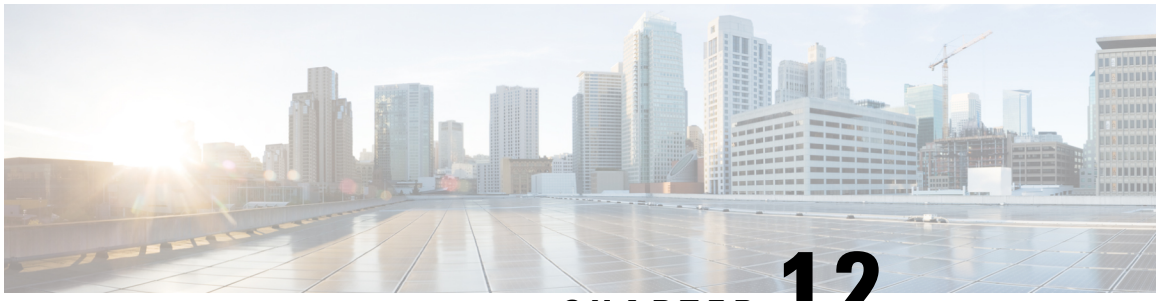
Auto-upgrade:Disabled

Auto-upgrade PM:Disabled
Attribute codes: B golden, P protect, S secure, A Anti Theft aware
```

Auto upgrade support for SC/MPA

In Cisco 8000 Series Routers, the auto upgrade on bootup path is being supported for new CPU less cards SC and MPA.

The RP and SC cards together form a domain in Active and Standby nodes. The respective domain lead (RP) is responsible to trigger the auto upgrade of respective SC cards.



CHAPTER 12

Release-specific Caveats and Workarounds

Some IOS XR releases have changes to the upgrade procedure due to new features and enhancements in these releases. The information in this section summarises the key changes. To view supported upgrade paths and caveats for any release upgrade path, use the **show install upgrade-matrix** commands.

- Release 25.3.1, on page 207
- Release 25.2.1, on page 209
- Release 7.10.1, on page 211
- Release 7.9.1, on page 213
- Release 7.7.2, on page 214
- Release 7.7.1, on page 214
- Release 7.5.2, Release 7.5.3, on page 215
- Release 7.5.1, Release 7.3.2, on page 216

Release 25.3.1

The caveats in 25.3.1 release are:

- CSCwo09365: FPDs not upgraded prior to install operation for upgrade to 25.2.1+ (Cisco 8000).
- CSCwo11635: Limit on Harddisk Files preserved in 8000-x86 upgrades/downgrades to/from versions >=25.2.1.
- CSCwq98264: No automatic recovery back to start software if unexpected reload occurs during install.
- CSCwq98275: **install package replace** and **install rollback** no longer supported for release upgrade/downgrade.

Table 41: Upgrade caveats

From	To	Bridge SMUs required (CSCwm77418)	Caveats
7.10.2	25.3.1	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
7.11.2 , 7.11.21	25.3.1	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635

From	To	Bridge SMUs required (CSCwm77418)	Caveats
24.1.2	25.3.1	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.2.2	25.3.1	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.2.20, 24.2.20x, 24.2.21	25.3.1	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.3.1	25.3.1	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.3.2	25.3.1	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.3.20, 24.3.30	25.3.1	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.4.1	25.3.1	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.4.2	25.3.1	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.1.1	25.3.1	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.1.2	25.3.1	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.2.1	25.3.1	No	CSCwq98264, CSCwq98275, CSCwo09365
25.2.15, 25.2.2	25.3.1	No	CSCwq98264, CSCwq98275, CSCwo09365

Table 42: Downgrade caveats

From	To	Target SMUs required (CSCwm77418)	Caveats
25.3.1	7.10.2	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.3.1	7.11.2 , 7.11.21	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.3.1	24.1.2	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.3.1	24.2.2	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635

From	To	Target SMUs required (CSCwm77418)	Caveats
25.3.1	24.2.20, 24.2.20x, 24.2.21	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.3.1	24.3.1	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.3.1	24.3.2	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.3.1	24.3.20, 24.3.30	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.3.1	24.4.1	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.3.1	24.4.2	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.3.1	25.1.1	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.3.1	25.1.2	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.3.1	25.2.1	No	CSCwq98264, CSCwq98275, CSCwo09365
25.3.1	25.2.15, 25.2.2	No	CSCwq98264, CSCwq98275, CSCwo09365

Release 25.2.1

The caveats in 25.2.1 release are:

- CSCwo09365: FPDs not upgraded prior to install operation for upgrade to 25.2.1+ (Cisco 8000).

  CSCwo11635: Limit on Harddisk Files preserved in 8000-x86 upgrades/downgrades to/from versions >=25.2.1.

- CSCwq98264: No automatic recovery back to start software if unexpected reload occurs during install.
- CSCwq98275: **install package replace** and **install rollback** no longer supported for release upgrade/downgrade.

For further information on these caveats, see Restrictions for disk encryption.

Table 43: Upgrade caveats

From	To	Bridge SMUs required (CSCwm77418)	Caveats
7.10.2	25.2.1+	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
7.11.2 , 7.11.21	25.2.1+	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.1.2	25.2.1+	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.2.11	25.2.1+	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.2.2	25.2.1+	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.2.20, 24.2.21	25.2.1+	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.3.1	25.2.1+	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.3.2	25.2.1+	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.3.20	25.2.1+	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.4.1	25.2.1+	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
24.4.2	25.2.1+	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.1.1	25.2.1+	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.1.2	25.2.1+	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635

Table 44: Downgrade caveats

From	To	Target SMUs required (CSCwm77418)	Caveats
25.2.1+	7.10.2	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.2.1+	7.11.2 , 7.11.21	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635

From	To	Target SMUs required (CSCwm77418)	Caveats
25.2.1+	24.1.2	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.2.1+	24.2.11	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.2.1+	24.2.2	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.2.1+	24.2.20, 24.2.21	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.2.1+	24.3.1	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.2.1+	24.3.2	Yes	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.2.1+	24.3.20	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.2.1+	24.4.1	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.2.1+	24.4.2	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.2.1+	25.1.1	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635
25.2.1+	25.1.2	No	CSCwq98264, CSCwq98275, CSCwo09365, CSCwo11635

Release 7.10.1

The following upgrade caveats are applicable for Release 7.10.1 and later:

Table 45: Upgrade Caveats

From	To	Bridge SMUs Required	Caveats
7.3.3	7.10.1 and later	Yes	1*, 2*, 3*
7.3.4	7.10.1 and later	Yes	1*, 2*, 3*
7.5.3	7.10.1 and later	None	1*
7.5.4	7.10.1 and later	None	1*
7.7.1	7.10.1 and later	None	1*

From	To	Bridge SMUs Required	Caveats
7.7.2	7.10.1 and later	None	1*
7.8.1	7.10.1 and later	None	1*
7.8.2	7.10.1 and later	None	1*
7.9.1	7.10.1 and later	None	1*
7.9.2	7.10.1 and later	None	1*

1*: You can't roll back using the **install rollback** command.

2*: Ensure that a reload bridging SMU (CSCwd71524) is installed.

3*: Ensure that you install the bridge SMU (CSCwd71524) manually because even if it's available inside the GISO that's replacing the existing GISO, this SMU doesn't get installed automatically.

Note CSCwd71524:

- When you upgrade from earlier than Release 7.10.1 to Release 7.10.1, system supports the installation process seemlessly.
- When you downgrade from Release 7.10.1, system preserves the present configuration and the install history from last transaction.

The following downgrade caveats are applicable for Release 7.10.1 and later:

Table 46: Downgrade Caveats

From	To	Bridge SMUs Required	Caveats
7.10.1 and later	7.3.3	Yes	C*
7.10.1 and later	7.3.4	Yes	C*
7.10.1 and later	7.5.3	Yes	***, A*, B*
7.10.1 and later	7.5.4	Yes	***, A*
7.10.1 and later	7.7.1	Yes	***, A*, B*
7.10.1 and later	7.7.2	Yes	***, A*, B*
7.10.1 and later	7.8.1	Yes	***, A*, B*
7.10.1 and later	7.8.2	Yes	***, A*, B*
7.10.1 and later	7.9.1	Yes	***
7.10.1 and later	7.9.2	Yes	***

***

- You don't need to run the **install commit** command after a downgrade operation because the operation is automatically committed.
- You can't roll back after a downgrade. To revert to the previous IOS XR previous version, replace or reimage to the relevant ISO.
- IOS XR configuration history is lost after a downgrade, but the router preserves the latest configuration.
- Install history from the last transaction is preserved after a downgrade operation.
- Downtime takes a longer time as the operation is performed through reimage.
- You can't downgrade using the **install package replace** command. Instead, use the **install replace** command to downgrade.
- Ensure that you reinstall third-party application once you complete the downgrade.
- PXE recovery is required if the image downgrading isn't bootable.
- You must re-install the *Type 6 masterkey* and reapply the configuration encrypted by it because they are lost after the downgrade.
- You must regenerate crypto keys and certificates after a downgrade.

A*: You can't downgrade to the base ISO. You can downgrade to a GISO containing the bridge SMU (CSCwd71524).

B*: You must recover the router through PXE if a power cycle occurs during the downgrade.

C*: One-step downgrade isn't supported. You must use either PXE/USB to downgrade or perform a two-step downgrade through Release 7.9.1 or Release 7.5.4. The first-hop downgrade to Release 7.9.1 or Release 7.5.4 still carries the same caveats.

Use the **show install upgrade-matrix running** command to view the caveats.

Release 7.9.1

The following caveats are applicable to Release 7.9.1 and later:

- CSCvy66646 (Hitless/Recommended SMU)—When you upgrade from releases earlier than 7.3.2 to release 7.8.2, we recommend that you install the `8000-version-CSCvy66646.tar` SMU from Cisco Software Download center and commit the install operation. Without this SMU, if you upgrade the router and if the router is reloaded due to any issue (excluding **install apply reload** command) before you commit the install operation, the system may prevent install operations in the future.
- CSCvw93597—If the **install package add** *pkg-name* command after **install package replace 8000-x64-7.9.1.iso** command fails when upgrading from release 7.3.15, rerun the **install package add** *pkg-name* command.
- CSCwc47306—The appmgr crashes continuously while downgrading from release 7.8.2 (with the healthcheck optional RPM) to releases earlier than 7.8.2. There is no impact to the upgrade operation.
- CSCwd59323—The counter-size value configured using **healthcheck metric** command is lost when the router is upgraded to release 7.8.2. This size indicates the buffer that stores the history of the counter value. Reconfigure the counter-size to a value in the range of 2 to 15 cadence snapshots.

- CSCwc47212—Configuration on breakout interface is lost after downgrading to releases earlier than 7.8.1 on 8202 router variants. Reapply the configuration.
- CSCwd30936—The `ema_server_sdr` process crashes after downgrading to releases earlier than 7.8.1. There is no workaround and no impact to the functionality.

Release 7.7.2

The following caveats are applicable for Release 7.7.2 and later:

- CSCvy66646 (Hitless/Recommended SMU)—When you upgrade from releases earlier than 7.3.2 to release 7.7.2, we recommend that you install the `8000-version-CSCvy66646.tar` SMU from Cisco Software Download center and commit the install operation. Without this SMU, if you upgrade the router and if the router is reloaded due to any issue (excluding **install apply reload** command) before you commit the install operation, the system may prevent install operations in the future.
- CSCvw93597—If the **install package add** *pkg-name* command after **install package replace 8000-x64-7.7.2.iso** command fails when upgrading from release 7.3.15, rerun the **install package add** *pkg-name* command.
- CSCwc47306—The appmgr crashes continuously while downgrading from release 7.7.2 (with the healthcheck optional RPM) to releases earlier than 7.7.2. There is no impact to the upgrade operation.
- CSCwb36889—When you upgrade from release 7.3.x to release 7.7.2, the line cards (LCs) may continue to be in the `BOOT HOLD` state, or the BIOS FPD may be in `NEED UPGD` state. This is an intermittent behavior and we recommend that you install the SMU on the 7.3.x image before upgrading to release 7.7.2.
- CSCwd37438—Upgrading from releases earlier than 7.7.2 leads to an additional silent reload due to BMC FPGA upgrade. This is specific to only 8201 and 8202 chassis. There is no impact to the upgrade operation. We recommend that you install the RPM before upgrading to release 7.7.2.

Release 7.7.1

The following upgrade caveats are applicable for Release 7.7.1 and later:

- CSCvy66646 (Hitless/Recommended SMU)—When you upgrade from releases earlier than 7.3.2 to releases 7.7.1, we recommend that you install the `8000-version-CSCvy66646.tar` SMU from Cisco Software Download center and commit the install operation. Without this SMU, if you upgrade the router and if the router is reloaded due to any issue (excluding **install apply reload** command) before you commit the install operation; the system may prevent install operations in the future.
- CSCvw93597—If the **install package add** *pkg-name* command after **install package replace** *iso-image* command fails when upgrading from release 7.3.15, rerun the **install package add** *pkg-name* command.
- CSCvz88814—The upgrade operation fails only in the following scenario:
  1. Upgrade from release 7.3.1 to release 7.7.1
  2. Downgrade from release 7.7.1 to release 7.3.1
  3. Upgrade again to release 7.7.1. The operation fails.

To avoid the failure when you upgrade after you downgrade the router, run the following commands in order:

1. **install package remove** *any optional package*
2. **install package abort all-since-apply**
3. **install replace** *iso-image*

Release 7.5.2, Release 7.5.3

The following caveats are applicable for Release 7.5.2 and Release 7.5.3:

- CSCvy66646 (Hitless/Recommended SMU)—When you upgrade from releases earlier than 7.3.2 to releases 7.5.2 or 7.5.3, we recommend that you install the `8000-version-CSCvy66646.tar` SMU from Cisco Software Download center and commit the install operation. Without this SMU, if you upgrade the router and if the router is reloaded due to any issue (excluding **install apply reload** command) before you commit the install operation; the system may prevent install operations in the future.
- CSCvw93597—If the **install package add** *pkg-name* command after **install package replace** *iso-image* command fails when upgrading from release 7.3.15. To solve the issue, rerun the **install package add** *pkg-name* command.
- CSCvz44123—An error message `ACCESS failure 'fail to get BiosGolden fpd info` displayed on the BIOS does not have a functional impact on the router. After the router is upgraded, the FPD shows the `CURRENT` state.

```
RP/0/RP0/CPU0:Feb 14 21:05:55.720 UTC: fpd_client[251]:
%PLATFORM-CPA_INTF_FPD-3-ACCESS_ERROR : Node
0/RP0/CPU0 FPD BiosGolden ACCESS failure 'fail to get BiosGolden fpd info'
RP/0/RP0/CPU0:Feb 14 21:05:55.738 UTC: fpd_client[251]:
%PLATFORM-CPA_INTF_FPD-3-ACCESS_ERROR : Node
0/RP0/CPU0 FPD Bios ACCESS failure 'fail to get Bios fpd info'
```

- CSCvz88814—The upgrade operation fails only in the following scenario:
  1. Upgrade from release 7.0.14, 7.2.1 or 7.3.1 to release 7.5.2 or 7.5.3
  2. Downgrade from release 7.5.2 or 7.5.3 to release 7.0.14, 7.2.1 or 7.3.1
  3. Upgrade again to release 7.5.2 or 7.5.3. The operation fails.

  To avoid the failure when you upgrade after you downgrade the router, run the following commands in order:

  1. **install package remove** *any optional package*
  2. **install package abort all-since-apply**
  3. **install replace** *iso-image*

Release 7.5.1, Release 7.3.2

The following caveats are applicable for Release 7.5.1 and Release 7.3.2:

- CSCvv17670—The issue with FPDs not upgraded on the line cards in release 7.0.14 with default auto FPD enabled. This issue is resolved in release 7.3.2.
- When auto FPD is enabled, the FPDs are automatically updated when a SMU or image changes, including an updated firmware revision. Although the FPD auto upgrade is enabled by default, when upgrading to release 7.5.1 or 7.3.2, we recommend that you run the **fpd auto-upgrade enable** command to avoid FPD upgrade failures.



CHAPTER 13

Setup and Upgrade Commands

This section serves as a reference to view the list of commands related to setting up and upgrading the router. Use this section to understand the command syntaxes, default values and sample command usage with output.

- Action Commands, on page 217
- Show Commands, on page 217

Action Commands

- clear configuration inconsistency
- install apply
- install commit
- install package
- install source
- install rollback
- install replace
- reload
- reload bootmedia

Show Commands

- show version
- show platform
- show install active
- show install committed
- show fpd package
- show hw-module fpd

- show interfaces (frame relay)
- show inventory (Cisco IOS XR 64-bit)
- show ipv4 interface
- show ipv6 interface
- show running-config
- show redundancy
- show media