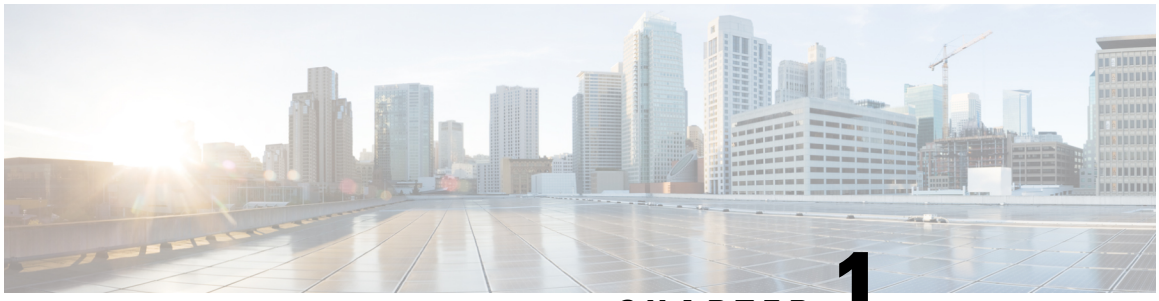
System Setup Guide for Cisco 8000 Series Routers, IOS XR Release 7.3.x

First Published: 2021-10-01
Last Modified: 2022-10-28

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

CONTENTS



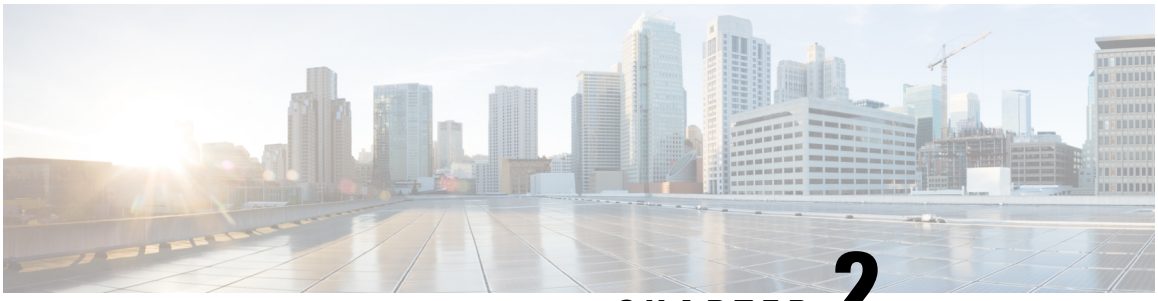
CHAPTER 1

New and Changed Feature Information

This table summarizes the new and changed feature information for the *System Setup and Software Installation Guide for Cisco 8000 Series Routers* .

New and Changed System Setup Features

Feature	Description	Changed in Release	Where Documented
Secure Zero Touch Provisioning with Removable Storage Device	With this release, you can securely and seamlessly provision network devices using USB.	Release 7.3.2	Secure ZTP with Removable Storage Device, on page 60
Recover System Using Console Port	With this feature, you can recover the router from disaster without having to reimage using iPXE or USB boot. The user data is securely erased before the router reloads.	Release 7.3.16	Recover System Using Console Port, on page 30
Conforming to US DOD Login Banner Standards	With this release, you can enable a login banner that conforms to the standards of the US DOD login banner.	Release 7.3.1	Create a User Profile, on page 28
Secure Zero Touch Provisioning	With this release, you can securely and seamlessly provision thousands of network devices accurately with out manual intervention.	Release 7.3.1	Securely Provision Your Network Devices, on page 51



CHAPTER 2

Get to Know Cisco 8000 Series Routers

Cisco 8000 series routers converge the service provider routing and massively scalable data centers (MSDC) switching portfolio. The routers run on XR 7 OS. The XR 7 OS provides significant architectural enhancements to Cisco IOS XR in these areas:

- **Modularity:** Decoupled hardware and software; modularized software with the flexibility to consume software packages based on requirement.
- **Programmability:** Model-driven APIs at all layers.
- **Manageability:** Simplified software management and installation based on Linux tools.
- Baseboard Management Controller, on page 3

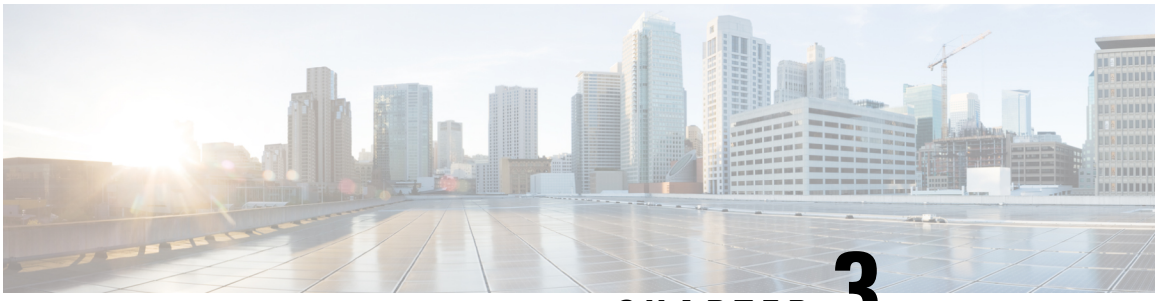
Baseboard Management Controller

Cisco 8000 series routers support Baseboard Management Controller (BMC) in the Route Processor (RP). BMC is a specialized service processor that communicates with the system through an independent ethernet connection. BMC operates in two modes:

- **Lights-on mode:** In this mode, RP CPU, which is independent and in parallel to BMC, is booted up. This mode provices access to remote console, ethernet management port, supervisory and environmetal device.
- **Lights-out mode:** In this mode, RP CPU is not booted up. This mode provides access to ethernet management port, troubleshooting and recovery, boot parameters for BIOS, supervisory and environmental device.

For information about how to configure the IP address for BMC, see Configure IP Address for Ethernet Port on BMC, on page 11.

This article helps you set up your Cisco 8000 series router. You will bring the router up, run a system health check, create user profiles, and assign user privileges.



CHAPTER 3

Bring-up the Cisco 8000 Series Router

Connect to the console port on a Route Processor (RP) of the router, and power ON the router. By default, this console port connects to the XR console. If necessary, after configuration, establish subsequent connections through the management port.

The following table shows the console settings:

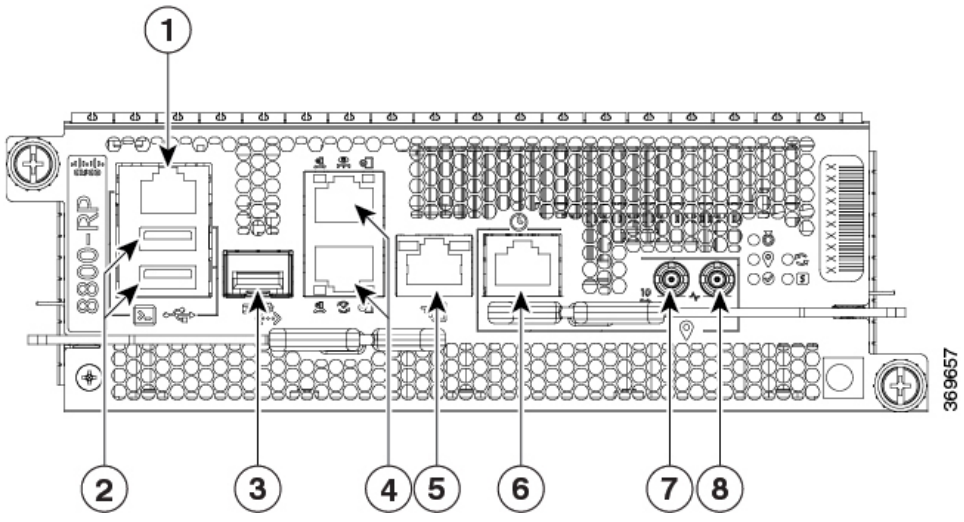
Table 1: Console Settings

Baud rate (in bps)	Parity	Stop bits	Data bits
115200	None	2	8

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

The router can be accessed using remote management protocols, such as SSH, Telnet, SCP and FTP. SSH is included in the software image by default, but telnet is not part of the software image. You must manually install the telnet optional package to use it.

Figure 1: Ports of the Route Processor



1	Console RS-232 Serial Port RJ45	5	SyncE BITS/DTI/J.211

2	USB Port Type-A (2 ports). Port A gets detected ahead of Port B Top: Port B Bottom: Port A	6	G.703 Time-of-Day (TOD)
3	Control Plane Expansion SFP/SFP+ port	7	Mini coaxial connector for 10 MHz, input, and output
4	Top: Management Ethernet (10/100/1000-Mbps) RJ-45 (Copper) port LAN shared between x86 (XR) and ARM11 Bottom: IEEE 1588 Precision Time Protocol (PTP)	8	Mini coaxial connector for 1 PPS, input, and output

After booting is complete, you must create a username and password. This credential is used to log on to the XR console, and get to the router prompt.

You can start or stop the console by using the following keyboard shortcuts:

- To start the console, press Ctrl + q.
- To stop the console, press Ctrl + s.

Note that by using Ctrl + s, the console output will be locked and you will need to initiate a Ctrl + q sequence to restore the console prompt.

The router completes the boot process using the pre-installed operating system (OS) image. If no image is available within the router, the router can be booted using iPXE boot or an external bootable USB drive.

Get to Know Cisco 8000 Series Routers

Cisco 8000 series routers converge the service provider routing and massively scalable data centers (MSDC) switching portfolio. The routers run on XR 7 OS. The XR 7 OS provides significant architectural enhancements to Cisco IOS XR in these areas:

- **Modularity:** Decoupled hardware and software; modularized software with the flexibility to consume software packages based on requirement.
- **Programmability:** Model-driven APIs at all layers.

- **Manageability:** Simplified software management and installation based on Linux tools.

Boot the Cisco 8000 Series Router Using Manual iPXE

Manually boot the router using iPXE if the router fails to boot when powered ON. An alternate method is to Boot the Cisco 8000 Series Router Using USB Drive.

iPXE is a pre-boot execution environment in the network card of the management interfaces. It works at the system firmware (UEFI) level of the router. iPXE boot re-images the system, boots the router in case of a boot failure, or in the absence of a valid bootable partition. iPXE downloads the ISO image, installs the image, and finally bootstraps inside the new installation.

iPXE acts as a bootloader. It provides the flexibility to choose the image that the system boots. The image is based on the Platform Identifier (PID), the serial number, or the management mac-address. iPXE is defined in the DHCP server configuration file.

You need a server running HTTPS, HTTP, or TFTP. Bring-up the PXE prompt using the following steps:

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

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

Step 1 Power ON the router.

Step 2 Press Esc or Del keys continuously (quick and repeated press and release) to pause the boot process, and get to the BIOS menu.

Step 3 Select `Boot Manager`, and then select `Built-in iPXE` option.

Step 4 When PXE boot starts reaching for a PXE server, press **Ctrl+B** keys to break into the PXE prompt.

Step 5 Add the following configuration for the router. This is required for the router to connect with the external server to download, and install the image. You can use HTTP, HTTPS or TFTP server.

Example:

```
iPXE> ifopen net0                    #Open the interface connecting outside world
iPXE> set net0/ip 10.0.0.2           #Configure the ip address of your router

iPXE> set net0/gateway 10.0.0.1      #configure the GW
iPXE> set net0/netmask 255.0.0.0     #Configure the Netmask
iPXE> ping 10.0.0.1                  #Check you can reach GW
iPXE> ping 192.0.2.0                 #check you can reach to your server running tftp or http or
https
iPXE> boot http://192.0.2.0/<directory-path>8000-x64.iso    #Copy the image on the http/https/tftp
server in any path and then point to download the image from there.
```

Note To rectify errors while typing the command, use **Ctrl+H** keys to delete a character.

If a PXE server is configured to run a DHCP server, it assigns an IP address to the Ethernet Management interface of the router. This provides a channel to download the image that is required to re-image a router in case of a boot failure.

```
Router#reload bootmedia network location all
Proceed with reload? [confirm]
```

Note Use the **force** option to perform an ungraceful reload of the specified location or hardware module. When **force** option is used along with the **all** location, the chassis undergoes an ungraceful reload. Use the **noprompt** option to avoid the prompt to confirm the operation. The **force** option is not recommended, and should not be used during regular operations.

Boot the Cisco 8000 Series Router Using USB Drive

Boot the router using USB drive if the router fails to boot when powered ON. An alternate method is to Boot the Cisco 8000 Series Router Using Manual iPXE.

Before you begin

Have access to a USB drive with a storage capacity that is between 8GB (min) and 32 GB (max). USB 2.0 and USB 3.0 are supported.

Note Use this procedure only on the active RP; the standby RP must either be powered OFF or removed from the chassis. After the active RP is installed with images from the USB drive, insert or power ON the standby RP as appropriate.

Step 1 Copy the bootable file to a USB disk.

A bootable USB drive is created by copying a compressed boot file into a USB drive. The USB drive becomes bootable after the contents of the compressed file are extracted.

Note If you are unable to boot from a USB drive, remove and insert the drive again. If the drive is inserted correctly, and still fails to read from the USB drive, check the contents of the USB on another system.

This task can be completed using Windows, Linux, or MAC operating systems available on your local machine.

a) Connect the USB drive to your local machine and format it with FAT32 or MS-DOS file system using the Windows Operating System or Apple MAC Disk Utility. To check if the disk is formatted as FAT32, right click on the USB disk, and view the properties.
b) Copy the compressed boot file in .zip format from the image file to the USB drive. This .zip file can be downloaded from the Cisco Software Download center.
c) Verify that the copy operation is successful. To verify, compare the file size at source and destination. Additionally, verify the MD5 checksum value.
d) Extract the contents of the compressed boot file by unzipping it inside the USB drive. This converts the USB drive to a bootable drive.

Note Extract the contents of the zipped file ("EFI" and "boot" directories) directly into the root folder of the USB 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 drive.

e) Eject the USB drive from your local machine.

Step 2 Use the bootable USB drive to boot the router or upgrade its image using one of the following methods:

Note Insert the USB drive in the USB port of the ACTIVE RP.

- **Boot menu**

  a. Insert the USB drive, and connect to the console.

  b. Power ON the router.

  c. Press Esc or Del to pause the boot process, and get the RP to the BIOS menu.

  d. Select `Boot Manager`, and then select the `USB` option from the boot menu.

```
Cisco BIOS Setup Utility - Copyright (C) 2019 Cisco Systems, Inc

Boot Override
UEFI: Micron_M600_MTFDDAT064MBF, Partition 4
UEFI: Built-in iPXE
URFI: Built-in Shell
URFI: Built-in Grub
UEFI: USB Flash Memory1.00, Partition 1
```

  The system boots the image from the USB drive, and installs the image onto the hard disk. The router boots from the hard disk after installation.

- **XR CLI**

  Use this method if you can access the XR prompt.

  Note The RP has two USB ports. If there is only one USB drive with a bootable image, insert it into any of the two USB ports. If there are two USB drives but only one has a bootable image, the choice of the USB port is negligent. However, if two USB drives are inserted simultaneously and both have a bootable image, the image in the lower USB port takes precedence.

  a. Insert the USB device in the RP.

  b. Access the XR prompt and run the command:

```
Router#reload bootmedia usb noprompt

Welcome to GRUB!!
Verifying (hd0,msdos1)/EFI/BOOT/grub.cfg...
(hd0,msdos1)/EFI/BOOT/grub.cfg verified using Pkcs7 signature.
Loading Kernel..
Verifying (loop)/boot/bzImage...
(loop)/boot/bzImage verified using attached signature.
Loading initrd..
Verifying (loop)/boot/initrd.img
```

  Use the **force** option to perform an ungraceful reload of the specified location or hardware module. When **force** option is used along with the **all** location, the chassis undergoes an ungraceful reload. Use the **noprompt** option to avoid the prompt to confirm the operation. The **force** option is not recommended, and should not be used during regular operations.

  The system boots the image from the USB and installs the image onto the hard disk. The router boots from the hard disk after installation.

Note Execute the `install commit` command before proceeding to the next install iteration, while performing cyclic upgrade and downgrade tests.

Configure the Management Port on the Cisco 8000 Series Router

To use the management port for system management and remote communication, you must configure an IP address and a subnet mask for the Management Ethernet interface.

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

Before you begin

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

Step 1 Configure a VRF.

Example:

```
Router#conf t
Router(config)#vrf <vrf-name>
Router(config-vrf)#exit
```

Step 2 Enter interface configuration mode for the management interface of the RP.

Example:

```
Router(config)#interface mgmtEth 0/RP0/CPU0/0
```

Step 3 Assign an IP address and a subnet mask to the interface.

Example:

```
Router(config-if)#ipv4 address 10.10.10.1/8
```

Step 4 Configure the Management Ethernet interface under the VRF.

Example:

```
Router(config-if)#vrf <vrf-name>
```

Step 5 Exit the management interface configuration mode.

Example:

```
Router(config-if)#exit
```

Step 6 Assign a virtual IP address and a subnet mask to the interface. The virtual address is primarily used for out-of-band management over the Management Ethernet interface.

Example:

```
Router(config)#ipv4 virtual address vrf <vrf-name> 10.10.10.1/8
```

Step 7 Place the interface in `UP` state.

Example:

```
Router(config)#no shutdown
```

Step 8 Specify the IP address of the default-gateway to configure a static route; this is used for communications with devices on other networks.

Example:

```
Router(config)#router static vrf <vrf-name> address-family ipv4 unicast 0.0.0.0/0 10.10.10.1
```

Step 9 Commit the configuration.

Example:

```
Router(config)#commit
```

Step 10 Connect to the management port to the ethernet network. With a terminal emulation program, establish a SSH or telnet connection to the management interface port using its IP address.

Configure IP Address for Ethernet Port on BMC

Baseboard Management Controller (BMC) is a component in the Route Processor (RP) that monitors bootup status and the health of hardware components using sensors. It communicates with the system through an independent connection. The independent connection is through a dedicated ethernet connection between the host and BMC. BMC also has an ethernet interface for connections external to the router. You can establish communication with this interface using REST or SSH services.

You can configure static IP or use DHCP for automatic IP assignment by DHCP server. For static IP assignment, connect to BMC console.

Note The Management Ethernet port is shared between XR and BMC. However, the IP address of BMC must be different from the XR interface, but in the same range.

To establish communication over ethernet (external to BMC and XR), configure static IP address on ethernet port 0 (eth0). IPv4 and IPv6 static IP addresses can be assigned. Modify the template in `/etc/systemd/network/00-bmc-eth0.network` with appropriate static IP address and gateway information. To modify the file, you must have root user privileges. Once modified, the system assigns the same IP address on eth0 ethernet device across BMC reloads.

Step 1 Switch to BMC console from the XR console.

Example:

```
Router#[ctrl] o
Phosphor OpenBMC (Phosphor OpenBMC Project Reference Distro) 0.1.0 ttyS4
```

Step 2 Set up a root username and password for BMC.

Example:

```
login: root
You are required to change your password immediately (administrator enforced)
New password:
Retype new password:
```

Step 3 Check that the BMC configuration file is available. If the file is unavailabke, then create one in the following template.

Example:

```
root:~# cat /etc/systemd/network/00-bmc-eth0.network
[Match]
Name=eth0
[Network]
DHCP=ipv4
LinkLocalAddressing=fallback
[DHCP]
ClientIdentifier=mac

# For static ip addresses replace above two sections with the following section
#[Network]
#Address=a.b.c.d/xy
#Gateway=a.b.p.q
```

Step 4 Modify the file using `Vi` text editor. Configure BMC with the network address and the gateway information.

Example:

```
vi /etc/systemd/network/00-bmc-eth0.network
```

Step 5 Save the file.

Step 6 View the content of the modified file.

Example:

```
root:~# cat /etc/systemd/network/00-bmc-eth0.network
[Match]
Name=eth0
#[Network]
#DHCP=ipv4
#LinkLocalAddressing=fallback
#[DHCP]
#ClientIdentifier=mac

# For static ip addresses replace above two sections with the following section
[Network]
Address=192.168.0.2/24
Gateway=192.168.0.1
```

Step 7 Reboot BMC using `reboot` Linux command for the configuration to take effect.

Step 8 After BMC reboots, verify that the static IP is present for Ethernet 0 device in BMC.

Example:

```
root:~# ifconfig eth0
Link encap:Ethernet  HWaddr 00:59:DC:16:A6:2E
inet addr:192.168.0.2  Bcast:192.168.0.1  Mask:255.255.0.0
inet6 addr: 2001:DB8:FFFF:FFFF:FFFF:FFFE:FFFF:FFFF Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

```
RX packets:1086 errors:0 dropped:0 overruns:0 frame:0
TX packets:205 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:185996 (181.6 KiB)  TX bytes:22383 (21.8 KiB)
Interrupt:20
```

Step 9 Verify connectivity to the external server.

Example:

```
root:~# ping -c 5 192.168.2.10
PING 192.168.2.10 (192.168.2.10): 56 data bytes
64 bytes from 192.168.2.10: seq=0 ttl=64 time=1.381 ms
64 bytes from 192.168.2.10: seq=1 ttl=64 time=0.881 ms
64 bytes from 192.168.2.10: seq=2 ttl=64 time=0.855 ms
64 bytes from 192.168.2.10: seq=3 ttl=64 time=0.865 ms
64 bytes from 192.168.2.10: seq=4 ttl=64 time=0.953 ms

--- 192.168.2.10 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0.855/0.987/1.381 ms
```

Note You can use Intelligent Platform Management Interface (IPMI) and REST interfaces to manage out-of-band BMC information.

For information about structure and usage examples for the REST interface, see OpenBMC REST API and OpenBMC REST cheat sheet.

By default, IPMI capability is disabled. For IPMI to function, you must enable `netipmi` using the following command:

```
netipmi_config.sh -s 1
```

For example, the following ipmi command displays sensors using BMC IP address:

```
root:~# ipmitool -H 192.0.2.0 -I lanplus -P 0penBmc sensor list
IBV             | 1031.000 | Volts | ok | 8917.000 | 9112.000 | 9307.000 | 1130.000 | 1171.000
 | 11972.000
VP1P2_CPU_C     | 1230.000 | Volts | ok | 1134.000 | 1158.000 | 1182.000 | 1256.000 | 1280.000
 | 1304.000
VP1P05_CPU      | 1060.000 | Volts | ok | 976.000  | 996.000  | 1018.000 | 1080.000 | 1102.000
 | 1122.000
VP0P6_VTT_MEM_C | 614.000  | Volts | ok | 558.000  | 570.000  | 582.000  | 618.000  | 630.000
  | 642.000
VP1P2_MGTAVTT   | 1210.000 | Volts | ok | 1092.000 | 1116.000 | 1140.000 | 1260.000 | 1284.000
 | 1308.000
VP1P0_MGTAVCC   | 1006.000 | Volts | ok | 910.000  | 930.000  | 950.000  | 1050.000 | 1070.000
 | 1090.000
VP3P3_OCXO      | 3339.000 | Volts | ok | 3003.000 | 3069.000 | 3135.000 | 3465.000 | 3531.000
 | 3597.000
P3_3V           | 3342.000 | Volts | ok | 3003.000 | 3069.000 | 3135.000 | 3465.000 | 3531.000
 | 3597.000
P2_5V           | 2528.000 | Volts | ok | 2273.000 | 2324.000 | 2375.000 | 2624.000 | 2675.000
 | 2723.000
VP1P8_OCXO      | 1814.000 | Volts | ok | 1638.000 | 1674.000 | 1710.000 | 1890.000 | 1926.000
 | 1962.000
P1_8V           | 1820.000 | Volts | ok | 1638.000 | 1674.000 | 1710.000 | 1890.000 | 1926.000
 | 1962.000
-------------------------    snipped   --------------------------
```

To disable netipmi, use the command:

```
root:~# netipmi_config.sh -s 0
```

Synchronize Router Clock with NTP Server

Synchronize the XR clock with that of an 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 The Cisco implementation of NTP does not support stratum 1 service.

Before you begin

Configure and connect to the management port.

Step 1 Enter the XR configuration mode.

Example:

```
Router#configure
```

Step 2 Synchronize the console clock with the specified sever.

Example:

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

The NTP source IP address can either be an IPv4 or an IPv6 address. For example:

IPv4:

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

IPv6:

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

Note The NTP server can also be reachable through a VRF if the Management Ethernet interface is in a VRF.

Step 3 Commit the configuration.

Example:

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

Step 4 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
```

Reload a Node on Cisco 8000 Series Router

Reload a specified location or the complete hardware module. This command when used with **all** option reloads the chassis. Reloading of a hardware module reloads all the locations on that card.

Use the **force** option to perform an ungraceful reload of the specified location or hardware module. When **force** option is used along with the **all** location, the chassis undergoes an ungraceful reload. Use the **noprompt** option to avoid the prompt to confirm the operation. The **force** option is not recommended, and should not be used during regular operations.

Step 1 Reload a specific location or the complete hardware module.

Example:

The following example shows reloading a specific location:

```
Router#reload location 0/RP1/CPU0
Proceed with reload? [confirm]
```

Example:

The following example shows reloading of the complete hardware module:

```
Router#reload location 0/RP1
Proceed with reload? [confirm]
```

Example:

The following example shows ungraceful reloading of a specific location:

```
Router#reload location 0/1/CPU0 force
Wed Sep 28 21:27:25.597 UTC
Attention! You have chosen to force reload this node.
This is an ungraceful operation and should only be used as a last resort.
Proceed with reload? [confirm]
```

Step 2 Verify that the node is reloaded.

Example:

```
Router#show platform
```

Note In Exec mode, `0/XXX/CPU0` denotes a specific location, and `0/XX` donotes the complete hardware module. For example, `0/1/CPU0` denotes the CPU0 location on module 1, `0/1` denotes the complete hardware module.

Note Ensure that you have applied the changes during a device upgrade.

In a multinode system, any node reloads that occur during a transaction that are not initiated as part of the install 'apply by reload' phase can result in the reloaded node being in BOOT HOLD state. The node continues to be in the BOOT HOLD state until the transaction is either committed or cancelled.

Shut Down a Node on Cisco 8000 Series Router

Shut down the complete hardware module for a specified location. To power on the hardware module for the specified location, use the **no** form of the command.

Step 1 Shut down the node using one of the two options:

- **Shut down from the configuration mode:**
  - **a.** Enter the XR configuration mode.

    Example:

    ```
    Router#config
    ```

  - **b.** Shut down the complete hardware module for a specified location. Route processors (RPs) cannot be shut down using this command.

    Example:

    ```
    Router(config)#hw-module shutdown location 0/1/CPU0
    ```

  - **c.** Commit the configuration.

    Example:

    ```
    Router(config)#commit
    Router:Sep 16 16:52:02.048 UTC: shelfmgr[270]: %PLATFORM-SHELFMGR-4-CARD_SHUTDOWN
    : Shutting down 0/1: User initiated shutdown from config
    ```

  Use **no hw-module shutdown location** <*location*> command to power on the hardware module for the specified location.

  ```
  Router(config)#no hw-module shutdown location 0/1/CPU0
  Router(config)#commit
  Router:Sep 16 16:52:43.851 UTC: shelfmgr[270]: %PLATFORM-SHELFMGR-4-CARD_RELOAD :
  Reloading 0/1: User initiated no-shutdown from config
  ```

  Note Under the configuration mode, the location `CPU0` denotes the complete hardware module.

- **Shut down from the Exec mode:**
  - **a.** Use the **force** option to perform an ungraceful reload of the specified location or hardware module. When **force** option is used along with the **all** location, the chassis undergoes an ungraceful reload. Use the **noprompt** option to avoid the prompt to confirm the operation. The **force** option is not recommended, and should not be used during regular operations.

Example: The following example shows the shut down of a specific location:

```
Router#shutdown location 0/1/CPU0
Proceed with shutdown? [confirm]
```

Example: The following example shows the shut down of the complete hardware module:

```
Router#shutdown location 0/1
Proceed with shutdown? [confirm]
```

Example: The following example shows the ungraceful shut down of a specific location:

```
Router#shutdown location 0/1/CPU0 force
Wed Sep 28 21:27:54.085 UTC
Attention! You have chosen to force shutdown this node.
This is an ungraceful operation and should only be used as a last resort.
Proceed with shutdown? [confirm]
```

Note In Exec mode, `0/XXX/CPU0` denotes a specific location, and `0/XX` donotes the complete hardware module. For example, `0/1/CPU0` denotes the CPU0 location on module 1, `0/1` denotes the complete hardware module.

b. Confirm to proceed with the shut down operation.

Step 2 Verify that the node is shutdown.

Example:

```
Router#show platform
```

Note A shut down operation on the hardware module of a specific card must not be followed by a boot or reload operation for any of the locations of the same card. A shut down operation for a particular hardware module is followed by a boot or reload operation for the same hardware module to power the module.

For example, the **shutdown location 0/RP1** operation must not be followed by **boot location 0/RP1/CPU0** or **reload location 0/RP1/CPU0** command. Use **boot location 0/RP1** to power it on or **reload location 0/RP1** command to reset the complete hardware module.

Boot a Node on Cisco 8000 Series Router

Boot the specified location or the complete hardware module in the system. Booting up a hardware module powers on all the locations on that card. Use the **noprompt** option to avoid the prompt to confirm the operation.

Step 1 Boot a specific location or the complete hardware module.

Example:

The following example shows booting up of a specific location:

```
Router#boot location 0/1/CPU0
Proceed with boot? [confirm]
```

Example:

The following example shows the booting up of the complete hardware module:

```
Router#boot location 0/1
Proceed with boot? [confirm]
```

Step 2 Confirm to proceed with the boot operation.

Step 3 Verify that the node is booted up.

Example:

```
Router#show platform
```

Note In Exec mode, `0/XXX/CPU0` denotes a specific location, and `0/XX` donotes the complete hardware module. For example, `0/1/CPU0` denotes the CPU0 location on module 1, `0/1` denotes the complete hardware module.

CHAPTER 4

Perform Preliminary Checks with Cisco 8000 Series Router

After successfully logging into the console, you must perform some preliminary checks to verify the correctness of the default setup. Correct any issues that arise before proceeding with further configurations.

Verify Software Version on Cisco 8000 Series Router

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

You can view the overview of the running software. This includes the following information:

- Image name and version
- User who built the image
- Time the image was built
- Build workspace
- Build host
- ISO label

Note If any modifications are made to the running software on the booted ISO, only the IOS XR version is displayed in the label field and not the label included in the ISO.

- Copyright information
- Hardware information

Display the version of the Cisco IOS XR software, and its various software components that are installed on the router.

```
Router#show version
Cisco IOS XR Software, Version 7.0.11 LNT
Copyright (c) 2013-2019 by Cisco Systems, Inc.

Build Information:
 Built By     : xyz
 Built On     : Sat Jun 29 22:45:27 2019
 Build Host   : iox-lnx-064
 Workspace    : ../7.0.11/8000/ws/
 Version      : 7.0.11
 Label        : 7.0.11

cisco 8000
System uptime is 41 minutes
```

Verify Status of Hardware Modules on Cisco 8000 Series Router

Hardware modules such as RPs, LCs, fan trays, and power modules are installed on the router. The firmware on various hardware components of the router must be compatible with the Cisco IOS XR image installed. Incompatibility may cause the router to malfunction. Verify that all hardware and firmware modules are installed correctly and are operational.

Before you begin

Ensure that all required hardware modules are installed on the router.

Step 1 View the status of the system.

Example:

```
Router#show platform
Node               Type                  State                 Config state
--------------------------------------------------------------------------------
0/RP0/CPU0         8201-SYS(Active)      IOS XR RUN             NSHUT
0/RP0              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 detected on the router.

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     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 result, verify that all hardware modules that are installed on the chassis are listed. If a module is not listed, it indicates that the module is malfunctioning, or is not installed properly. Remove and reinstall that hardware module.

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

- FPD Device—Name of the hardware component, such as IO FPGA, IM FPGA, or BIOS

  Note Golden FPDs are not field upgradable.

- Status—Upgrade status of the firmware. The different states are:

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 newer 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.

- Running—Current version of the firmware running on the FPD
- Programd—Version of the FPD programmed on the module

Step 3 If necessary, upgrade the required firmware.

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

Step 4 After the modules are upgraded verify the status of the modules.

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     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 show that a reload is required.

Step 5 Reload the individual nodes that required an upgrade.

Example:

```
Router#reload location <node-location>
```

Note Use the **force** option to perform an ungraceful reload of the specified location or hardware module. When **force** option is used along with the **all** location, the chassis undergoes an ungraceful reload. Use the **noprompt** option to avoid the prompt to confirm the operation. The **force** option is not recommended, and should not be used during regular operations.

By default the system requests recovery reload, when the system detects fault. However, if you want to prevent the recovery reload for debugging, use the **hw-module reset auto disable** command to disable an auto reset mechanism.

```
Router(config)#hw-module reset auto disable location 0/RP1/CPU0
```

If you want to re-enable the recovery reload, use the **no hw-module reset auto disable location 0/RP1/CPU0** command.

Step 6 Verify that all nodes that required an upgrade show 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     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  0.24      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
```

Verify Interface Status on the Cisco 8000 Series Router

After the router has booted, all available interfaces must be discovered by the system. If interfaces are not discovered, it might indicate a malfunction in the unit.

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. Verify that the total number of interfaces 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 linecards displayed in **show platform** command.

Verify Node Status on Cisco 8000 Series Router

Each card on the router represents a node.

Verify the operational status of the node.

Example:

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

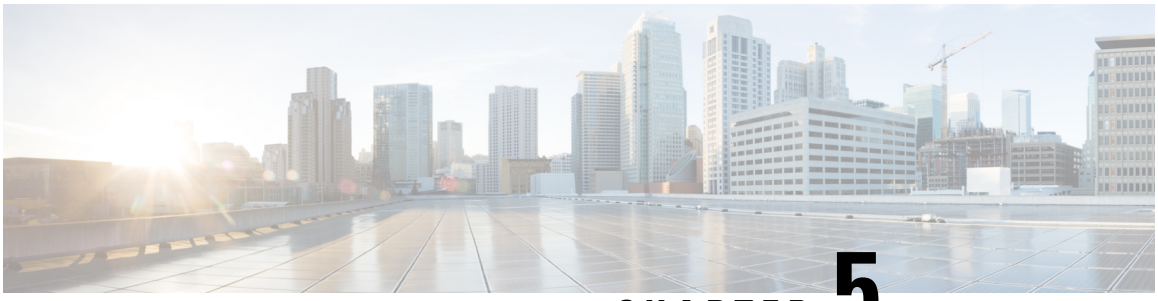
Displays the status of nodes present in the chassis.

Verify that the software state of all RPs, LCs, and the hardware state of FCs, FTs, and power modules are listed, and their state is `OPERATIONAL`. This indicates that the XR console is operational on the cards.

The platform states are described in the following table:

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

Card Type	State	Description
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
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



CHAPTER 5

Create Users and Assign Privileges on the Cisco 8000 Series Router

Users are 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

XR has its AAA separate from Linux. XR AAA is the primary AAA system. A user created through XR can log in directly to the EXEC prompt when connected to the router. A user created through Linux can connect to the router, but arrive at the bash prompt. The user must log in to XR explicitly in order to access the XR EXEC prompt.

Configure AAA authorization to restrict users from uncontrolled access. If AAA authorization is not configured, the command and data rules associated to the groups that are assigned to the user are bypassed. 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. In order to avoid granting uncontrolled access, enable AAA authorization 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*.

Create a User Profile

Table 2: Feature History Table

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

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

In this task, you create a user, `user1`, password for this user, `pw123`, and assign the user to a group `root-lr`.

Step 1 Enter the XR configuration mode.

Example:

```
Router#config
```

Step 2 Create a new user.

Example:

```
Router(config)#username user1
```

Step 3 Create a password for the new user.

Example:

```
Router(config-un)#password pw123
```

Step 4 Assign the user to group `root-lr`.

Example:

```
Router(config-un)#group root-lr
```

All users have `read` privileges. However, users can be assigned to `root-lr` usergroup. These users inherit the `write` privileges where users can create configurations, create new users, and so on.

Step 5 (Optional) You can enable the display of the US Department of Defense (DOD)-approved login banner. The banner is displayed before granting access to devices. The banner also ensures privacy and security that is consistent with applicable federal laws. In addition, the system keeps track of logins, right from the system boot, or as soon as the user profile is created.

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

Enable or disable the login banner using these commands:

Example:

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

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

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

Step 6 Commit the configuration.

Example:

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

What to do next

With the router set up, you can manage your system, install software packages, and configure your network.

Create a User Group

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.

In this task, you create a group name, `group1`, and assign a user, `user1` to this group.

Before you begin

Create a user profile. See Create a User Profile, on page 28.

Step 1 Enter the XR configuration mode.

Example:

```
Router#config
```

Step 2 Create a new user group, `group1`.

Example:

```
Router#(config)#group group1
```

Step 3 Specify the name of the user, `user1` to assign to this user group.

Example:

```
Router#(config-GRP)#username user1
```

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

Step 4 Commit the configuration.

Example:

```
Router#commit
```

What to do next

With the router set up, you can manage your system, install software packages, and configure your network.

Recover System Using Console Port

Table 3: Feature History Table

Feature name	Release Information	Feature Description
Recover System Using Console Port	Release 7.3.16	With this feature, you can recover the router from disaster without having to reimage using iPXE or USB boot. The user data is securely erased before the router reloads.

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.

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
- User files

Note The data on the line card is not erased.

This feature is disabled by default. Since the router can be recovered through the console, it is crucial to secure the physical access and the console.

The following steps show the process to recover the system in case of a disaster.

Before you begin

Prepare the system with the following requirements:

- Ensure you have administrator privileges.
- Enter the XR configuration mode. Enable the system recovery using console port.

  ```
  Router(config)#system recovery
  ```

  With this command, the functionality to recover the router is enabled. The logs are stored at `/var/log/system_recovery_logs/` location.

Note To disable this feature, use the **no** form of command.

```
Router(config)#no system recovery
```

Step 1 Power cycle the router using an external power cycler.

Step 2 Press `ESC` key and hold both active and standby RPs (RP0 and RP1) in BIOS.

This procedure must be executed on each RP individually on a distributed system.

Step 3 Boot on the standby RP. Press `ESC` key to enter the GRUB (bootstrap program) menu.

Step 4 Select the **IOS-XR-Recovery** option from the menu.

The RP boots in the recovery mode, clears generated files, and reboots.

Step 5 Hold the standby RP in BIOS prompt and initiate the recovery on the active RP.

The active RP boots up and the login prompt appears.

Step 6 Boot the standby RP.

After the system boots up, the syslog displays the status of the recovery operation. If the recovery operation fails, the system comes up to an inconsistent state. Power cycle and retry the recovery. If the router recovery is successful, configure the credentials to log in to the router with the preexisting image.

Note 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.

CHAPTER 6

Provision Network Devices using Zero Touch Provisioning

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.

Learn about Zero Touch Provisioning

ZTP allows you to provision the network device with day 0 configurations and supports both management ports and data ports.

Note 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. See Zero Touch Provisioning on a Fresh Boot of a Router, on page 34
- **Manual Invocation:** Use this method when you want to forcefully initiate ZTP on a fully configured device. See Manual ZTP Invocation, on page 46.

When to use Zero Touch Provisioning: Use Zero Touch Provisioning when the devices are in a secured network, but in an insecure network, we recommend you to Securely Provision Your Network Devices, on page 51.

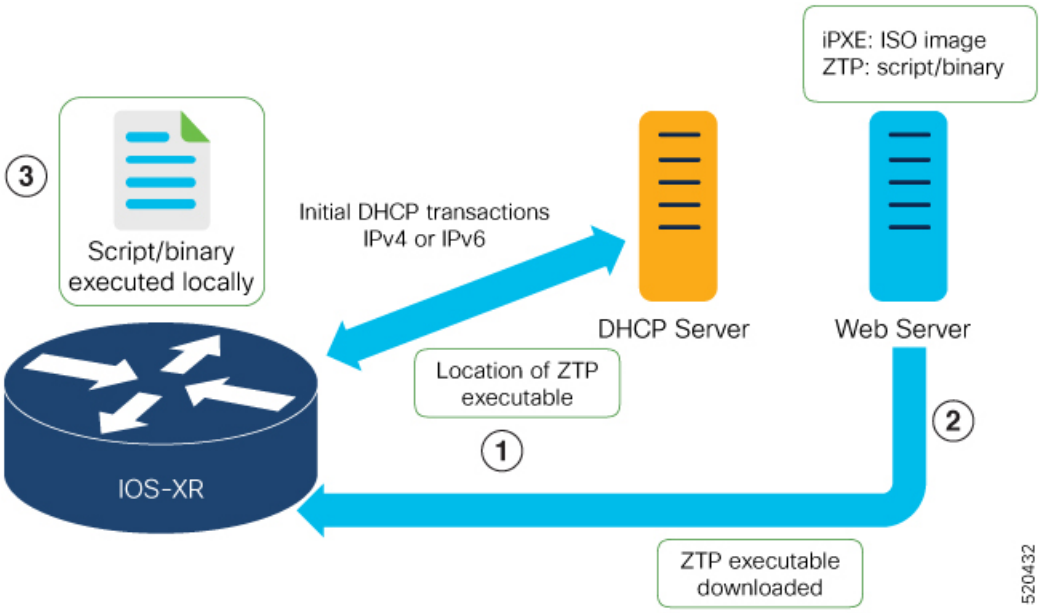
Zero Touch Provisioning on a Fresh Boot of a Router

When you boot the device, the ZTP process initiates automatically if the device does not have a prior configuration.

Fresh Boot Using DHCP

When you boot the device, the ZTP process initiates automatically if the device does not have a prior configuration. During the process, the router receives the details of the configuration file from the DHCP server.

This image depicts the high-level work flow of the ZTP process:



The ZTP process initiates when you boot the network-device with an IOS-XR image. The process starts only on the device that doesn't have a prior configuration.

Here is the high-level work flow of the ZTP process for the Fresh boot:

1. ZTP sends DHCP request to fetch the ZTP configuration file or user script. To help the Bootstrap server uniquely identify the device, ZTP sends below DHCP option
   - DHCP(v4/v6) client-id=Serial Number
   - DHCPv4 option 124: Vendor, Platform, Serial-Number
   - DHCPv6 option 16: Vendor, Platform, Serial-Number

   The following is the default sequential flow of the ZTP process:
   - ZTP sends IPv4 DHCP request first on all the management port. In case there is a failure, then ZTP sends IPv6 DHCP request on all the management port.
   - ZTP sends IPv4 DHCP request first on all the data port. In case there is a failure, then ZTP sends IPv6 DHCP request on all the data port.

     The default sequential flow is defined in configuration file and you can modify the sequence using the configuration file.

2. DHCP server identifies the device and responds with DHCP response using one of the following options:

   DHCP server should be configured to respond with the DHCP options.
   - DHCPv4 using BOOTP filename to supply script/config location.
   - DHCPv4 using Option 67 (bootfile-name) to supply script/config location.
   - DHCPv6 using Option 59 (OPT_BOOTFILE_URL) to supply script/config location

3. The network device downloads the file from the web server using the URI location that is provided in the DHCP response.
4. The device receives a configuration file or script file from the HTTP server.

Note
- If the downloaded file content starts with !! IOS XR it is considered as a configuration file.
- If the downloaded file content starts with #! /bin/bash, #! /bin/sh or #!/usr/bin/python it is considered as a script file.

5. The device applies the configuration file or executes the script or binary in the default bash shell.
6. The Network device is now up and running.

Build your Configuration File

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

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

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

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

The following is the sample configuration file:

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

hostname ios
alias exec al show alarms brief system active

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

Create User Script

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

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

ZTP Shell Utilities

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

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

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

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

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

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

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

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

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

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

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

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

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

- **xrreplace_with_extra_auth**—Used to apply XR configuration replace in XR namespace via a file The **xrreplace_with_extra_auth** API is used when configurations that require additional authentication to be applied such as alias, flex groups

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

ZTP Helper Python Library

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

ZtpHelpers Class Methods

Following are utility methods of the `ZtpHelpers` class:

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

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

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

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

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

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

```
Class Method to fetch the network namespace filepath
           associated with a PID or name
           :param cls: Reference to the class ZtpHelpers
           :param nspath: optional network namespace associated name
           :param nspid: optional network namespace associate PID
           :type nspath: str
           :type nspid: int
           :return: Return the complete file path
           :rtype:  str
```

- `toggle_debug(self, enable):`

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

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

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

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

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

- `setup_syslog(self):`

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

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

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

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

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

           :param reason: Reason for the config commit.
                          Will show up in the output of:
                          "show configuration commit list detail"
           :type filename: str
           :type reason: str
           :return: Dictionary specifying the effect of the config change
                    { 'status' : 'error/success', 'output': 'exec command based on
status'}
                    In case of Error:  'output' = 'show configuration failed'
                    In case of Success: 'output' = 'show configuration commit changes
 last 1'
           :rtype: dict
```

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

```
Apply Configuration to XR using  a single line string
           :param cmd: Single line string representing an XR config command
           :param reason: Reason for the config commit.
                          Will show up in the output of:
```

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

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

```
  Replace XR Configuration using a file

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

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

Example

The following shows the sample script in phyton

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

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


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


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

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

```
###### Debugs Disabled ######


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

Building configuration...
{'output': ['!! IOS XR Configuration version = 6.2.1.21I',
            '!! Last configuration change at Sat Dec 17 04:23:25 2016 by UNKNOWN',
            '!',
            'hostname customer2',
            'username root',
            'group root-lr',
            'group cisco-support',
            'secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1',
            '!',
            'username noc',
            'group root-lr',
            'group cisco-support',
            'secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1',
            '!',
            'username netops',
            'group root-lr',
            'group cisco-support',
            'secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1',
            '!',
            'username netops2',
            'group root-lr',
            'group cisco-support',
            'secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1',
            '!',
            'username netops3',
            'group root-lr',
            'group cisco-support',
            'secret 5 $1$7kTu$zjrgqbgW08vEXsYzUycXw1',
            '!',
            'cdp',
            'service cli interactive disable',
            'interface MgmtEth0/RP0/CPU0/0',
            'ipv4 address 11.11.11.59 255.255.255.0',
            '!',
            'interface TenGigE0/0/0/0/24',
            'shutdown',
            '!',
            'interface TenGigE0/0/0/0/25',
            'shutdown',
            '!',

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

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

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

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

Building configuration...
```

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

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

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

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

Set Up DHCP Server for ZTP

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

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

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

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

Example

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

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

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

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

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

The DHCP server responds with the following DHCP options:

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

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

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

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

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

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

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

    }

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

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


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

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


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

        }
      }


    ddns-hostname "cisco-local";
    option routers 11.11.11.2;
  }
}
```

Important In Cisco IOS XR Release 7.3.1 and earlier, the system accepts the device sending **user-class = "exr-config"**; however starting Cisco IOS XR Release 7.3.2 and later, you must use only **user-class = "xr-config"**.

In Cisco IOS XR Release 7.3.2 and later, use:

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

Also, when upgrading from any release that is Cisco IOS XR Release 7.3.1 or earlier to Cisco IOS XR Release 7.3.2 or later release, use the following:

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

Authentication on Data Ports

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

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

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

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

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

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

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

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

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

```
 #valid md5
 option CISCO-XR-CONFIG.md5sum "90fd845ac82c77f834d57a034658d0f0";
 if option dhcp6.user-class = 00:04:69:50:58:45 {
  option dhcp6.bootfile-url "http://[2001:1851:c632:1::1]/cisco-2/image.iso";
 }
 else {
   #option dhcp6.bootfile-url "http://[2001:1851:c632:1::1]/cisco-2/cisco-mini-x.iso.sh";
   option dhcp6.bootfile-url "http://[2001:1851:c632:1::1]/cisco-2/ztp.cfg";
  }
}
```

Manual ZTP Invocation

You can ZTP manually through Command Line Interface. This method is Ideal for verifying the ZTP configuration without a reboot. This manual way helps you to provision the router in stages. If you would like to invoke a ZTP on an interfaces (data ports or management port), you don't have to bring up and configure the interface first.

You can execute the ztp initiate command, even if the interface is down, ZTP script will bring it up and invoke dhclient. So ZTP could run over all interfaces no matter it is up or down.

Use the following commands to manually execute the ZTP commands to force ZTP to run over more interfaces:

- **ztp initiate** — Invokes a new ZTP DHCP session. Logs can be found in **/disk0:/ztp/ztp.log**.

  Configuration Example:

  ```
  Router#ztp initiate debug verbose interface HundredGigE 0/0/0/24
  Invoke ZTP? (this may change your configuration) [confirm] [y/n] :
  ```

- **ztp terminate** —Terminates any ZTP session in progress.

  Configuration Example:

  ```
  Router #ztp terminate verbose
  Mon Oct 10 16:52:38.507 UTC
  Terminate ZTP? (this may leave your system in a partially configured state) [confirm]
  [y/n] :y
  ZTP terminated
  ```

- **ztp enable** —Enables the ZTP at boot.

  Configuration Example:

  ```
  Router#ztp enable
  Fri Jul 12 16:09:02.154 UTC
  Enable ZTP? [confirm] [y/n] :y
  ZTP Enabled.
  ```

- **ztp disable** —Disables the ZTP at boot.

  Configuration Example:

  ```
  Router#ztp disable
  Fri Jul 12 16:07:18.491 UTC
  Disable ZTP? [confirm] [y/n] :y
  ZTP Disabled.
  Run ZTP enable to run ZTP again.
  ```

- **ztp clean** —Removes only the ZTP state files.

  Configuration Example:

```
Router#ztp clean verbose
Mon Oct 10 17:03:43.581 UTC
Remove all ZTP temporary files and logs? [confirm] [y/n] :y
All ZTP files have been removed.
If you now wish ZTP to run again from boot, do 'conf t/commit replace' followed by
reload.
```

The log file ztp.log is saved in **/var/log** folder, and a copy of log file is available at **/disk0:/ztp/ztp.log** location using a soft link. However, executing **ztp clean** clears files saved on disk and not on **/var/log** folder where current ZTP logs are saved. In order to have a log from current ZTP run, you must manually clear the ZTP log file from **/var/log/** folder.

Configuration

This task shows the most common use case of manual ZTP invocation: invoke ZTP.

1. Invoke DHCP sessions on all data ports which are up or could be brought up. ZTP runs in the background. Use show logging or look at /disk0:/ztp/ztp.log to check progress.

Configuration Example:

```
Router# ztp initiate dataport
```

Configure ZTP BootScript

If you want to hard code a script to be executed every boot, configure the following.

```
Router#configure
Router(config)#ztp bootscript /disk0:/myscript
Router(config)#commit
```

The above configuration will wait for the first data-plane interface to be configured and then wait an additional minute for the management interface to be configured with an IP address, to ensure that we have connectivity in the third party namespace for applications to use. If the delay is not desired, use:

```
Router#configure
Router(config)#ztp bootscript preip /disk0:/myscript
Router(config)#commit
```

Note When the above command is first configured, you will be prompted if you wish to invoke it now. The prompt helps with testing.

This is the example content of **/disk0:/myscript**:

```
#!/bin/bash
exec &> /dev/console # send logs to console
source /pkg/bin/ztp_helper.sh

# If we want to only run one time:
xrcmd "show running" | grep -q myhostname
if [[ $? -eq 0 ]]; then
    echo Already configured
```

```
fi

# Set the hostname
cat >/tmp/config <<%%
!! XR config example
hostname myhostname
%%
xrapply /tmp/config

#
# Force an invoke of ZTP again. If there was a username normally it would not run. This
forces it.
# Kill off ztp if it is running already and suppress errors to the console when ztp runs
below and
# cleans up xrcmd that invokes it. ztp will continue to run however.
#
xrcmd "ztp terminate noprompt" 2>/dev/null
xrcmd "ztp initiate noprompt" 2>/dev/null
```

Customize the ZTP Configurable Options

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
```

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

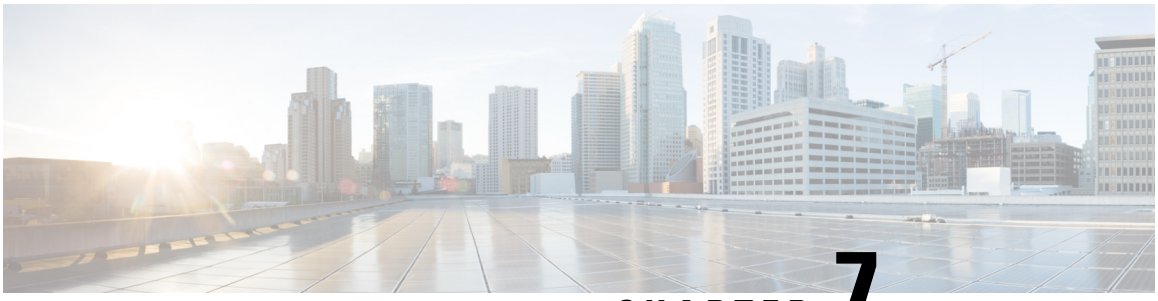
```
Router#ztp enable
Fri Jul 12 16:09:02.154 UTC
Enable ZTP? [confirm] [y/n] :y
ZTP Enabled.
```

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.
```



CHAPTER 7

Securely Provision Your Network Devices

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

Table 4: Feature History Table

Feature	Release Information	Feature Description
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.

The following are the topics covered in this chapter:

On board Devices Using Three-Step Validation

The 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 on board the remote devices securely:

1. **Router Validation**: The ZTP server authenticates the router before providing bootstrapping data using the Trust Anchor Certificate (SUDI certificate). Ensure that you have preinstalled the CA certificate chain for Cisco, as this is a prerequisite for the Cisco CA on ZTP server to verify the client/router SUDI certificates. The required certificates are:
   - subject=O = Cisco, CN = ACT2 SUDI CA
   - subject=O = Cisco Systems, CN = Cisco Root CA 2048
   - subject=CN = High Assurance SUDI CA, O = Cisco
   - subject=O = Cisco, CN = Cisco Root CA 2099
2. **Server Validation**: The router device in turn validates the ZTP server to make sure that the on board 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. See Secure ZTP Components , on page 52.
3. **Artifact Validation**: The configuration validates the bootstrapping data or artifact that is received from the ZTP server.

Secure ZTP Components

Let's first understand the components required for secure ZTP.

Table 5: Components used in Secure ZTP

Components	Description
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.

Components	Description
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 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.
	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. See Build your Configuration File, on page 36.
	  • Python library: Includes IOS XR CLI (show commands and configuration commands) and YANG-XML (ncclient, native Netconf client).
	  • BASH library: Includes IOS XR CLI show commands, configuration commands
	• Bootstrapping Data

Components	Description
Bootstrapping Data	

Components	Description

	Bootstrapping data 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`

Components	Description
	• **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==" }] } }`

Components	Description
	• **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==" } }`

Components	Description
	• **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.

Components	Description

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 Zero Touch Provisioning

When you boot the device, the secure ZTP process initiates automatically if the device does not have a prior configuration.

During the process, the router verifies the list of sources and receives the information of the configuration file accordingly. The following are the sources that can provide the configuration file information.

- Removable storage: A directly attached removable storage device, for example, USB flash drive.
- DHCP server

The section covers the following topics:

Secure ZTP with Removable Storage Device

A Removable storage device such as a USB drive is an untrusted source of bootstrapping data. So, the onboarding information present in the removable storage device must always be signed.

Whenever the data is signed, it's mandatory that the Owner Certificate and Ownership Voucher must also be available. The removable storage device must contain the following three artifacts. For more information on the three artifacts, see Secure ZTP Components , on page 52.

- Conveyed Information
- Owner Certificate
- Ownership Voucher

This section covers the following topics:

Prepare Removable Storage Device to Provision Secure ZTP

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

Before performing the following tasks, ensure to enable secure ZTP on the router using the **ztp secure-mode enable** command and then reload the router.

1. 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"
      }
   ```

2. Copy the following data to the removable storage device in the **EN9** directory in its root:
   - Conveyed information: Conveyed information must be named as `conveyed-information.cms` and must contain only the onboarding information and not the redirect information.

- Onboarding Information: The conveyed information consists of the following onboarding information:
  - Cisco IOS XR software images: You can download Cisco images, SMU, and patches using the Cisco Support & Downloads page.
    a. Click **Routers** and select the product that you want to download the image for.
    b. On the product home page, select the required Product model from the **Downloads** tab.
    c. From the **Software Available For This Product** page, download the required Cisco image, SMU, and patches.
  - ZTP scripts that include IOS XR configurations, pre, and post configuration scripts.

    During the secure ZTP process, secure ZTP executes the scripts to provision the router. You can build your script using one of the following methods:
    a. Python library: Includes IOS XR CLI (show commands and configuration commands) and YANG-XML (`ncclient`, `native Netconf client`).
    b. BASH library: Includes IOS XR CLI show commands, configuration commands.

    See Build your Configuration File, on page 36.

- Owner certificate: The owner certificate must be named as `owner-certificate.cms`.
- Ownership vouchers: The ownership vouchers must be named as `ownership-voucher.vcj`.

The artifacts must be stored inside the subdirectory named after the RP serial number of the router.The following example shows a directory structure for the router with RP serial number `FOC2202R293` containing all three artifacts:

```
EN9
└── FOC2202R293
    └── bootstrapping-data
        ├── conveyed-information.cms
        ├── owner-certificate.cms
        └── ownership-voucher.vcj
```

3. Plug in the removable storage device into the router.
4. Power on the router.

How Does Secure ZTP Work with Removable Storage Device?

Before you begin, complete the task to prepare the removable storage device. See Prepare Removable Storage Device to Provision Secure ZTP, on page 60.

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

1. 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.
2. 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. For more information, see Customize the ZTP Configurable Options, on page 48.

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
```

3. Secure ZTP checks for a removable storage device on the router. If the removable storage device isn't available, the secure ZTP process moves to the next fetcher as defined in the fetcher priority of the `ztp.ini`file.
4. If a removable storage device is available, the router scans for the `EN9` directory in the root of the removable storage device.

   If the `EN9` directory isn't available, the secure ZTP process moves to the next fetcher as defined in the fetcher priority of the `ztp.ini`file.
5. **Artifact Validation**:

   The router validates the artifacts received from the removable storage device.

   a. The router validates the ownership voucher and extracts the `pinned-domain-cert` node, an X.509 certificate from the ownership voucher to verify the owner certificate.

   b. The router authenticates the owner certificate by performing the X.509 certificate path verification process on the trusted certificate.

   c. Finally, the router verifies whether the conveyed information artifact is signed by the validated owner certificate.
6. **Provision the router**:

   a. The device first processes the boot image information.

   b. Executes the preconfiguration script and then commits the initial configuration.

   c. Execute the post configuration script.
7. After the onboarding process is completed, router is operational.

Note If there is a failure in any of the steps, the secure ZTP process moves to the next fetcher as defined in the fetcher priority of the `ztp.ini` file.

Secure ZTP with 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 section covers the following topics:

Initial Set Up for Secure ZTP

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

1. Contact Cisco Support to obtain a voucher. Provide the following details to request for ownership voucher certificate:
   - Pinned Domain Certificate: A trusted digital certificate issued by the Certificate Authority (CA) and pinned by the operator.
   - 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"
    }
```

2. 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 your Configuration File, on page 36.
     - Python library: Includes IOS XR CLI (show commands and configuration commands) and YANG-XML (`ncclient`, `native Netconf client`).
     - 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
3. 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>]".`

4. Power on the router.
5. Enable the secure ZTP option on the onboarding device. Execute the following command on your router to enable secure ZTP:

```
Router# ztp secure-mode enable
```

How Does Secure ZTP Work?

Before you begin, ensure that you configure the network with the DHCP and ZTP server. See Initial Set Up for Secure ZTP, on page 63.

1. 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.

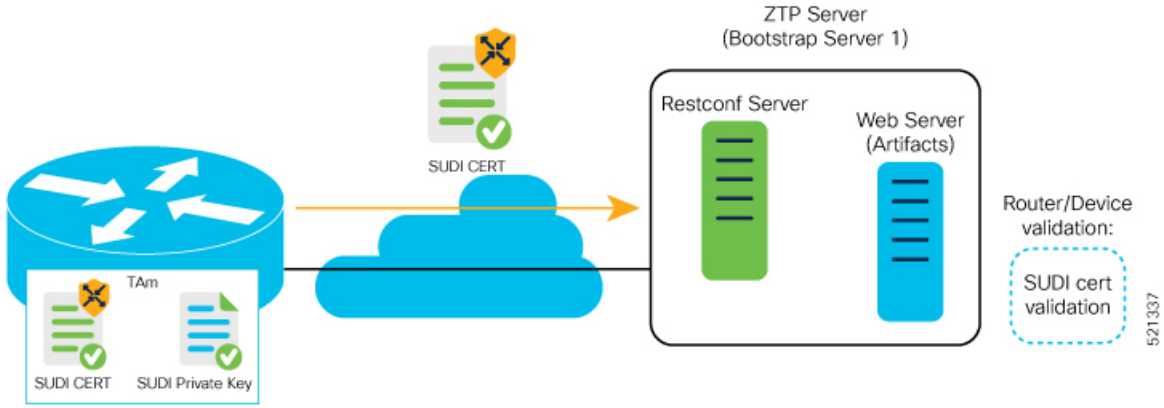
2. **DHCP discovery**:
   a. The router initiates a DHCP request to the DHCP server.
   b. The DHCP server responds with a DHCPv4 143 address option (for IPv4 addressing) or a DHCPv6 136 option (for IPv6 addressing). In addition, URLs to access bootstrap servers for further configuration is also listed.

Figure 2: DHCP discovery

3. **Router validation**:

   a. 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 3: Router Validation



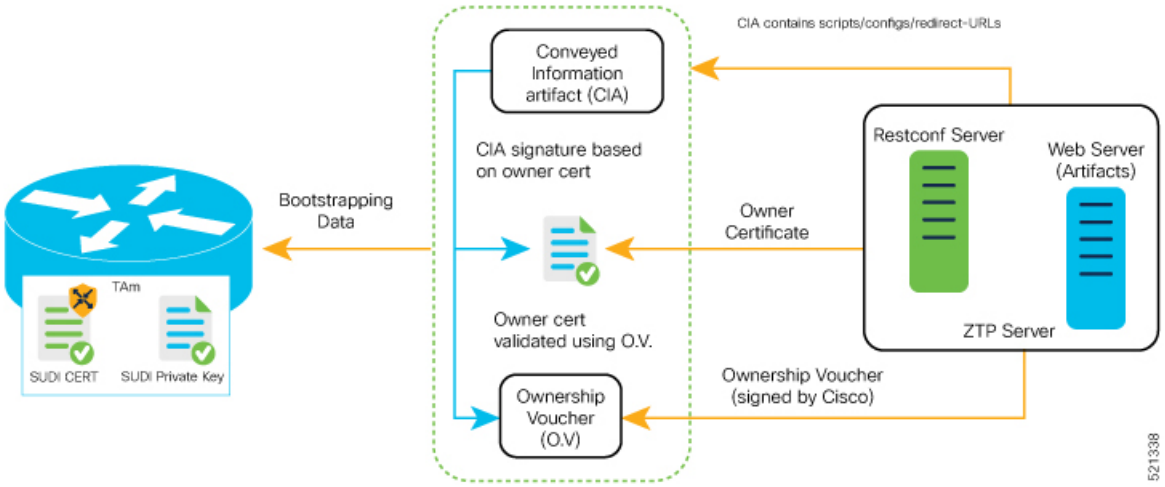
   b. The RESTCONF or HTTPs server verifies the received SUDI certificate with the public certificate that it contains. Cisco issues the public certificate to ensure that the onboarding device is an authorized Cisco device.

   c. After the onboarding device is authenticated, the web server sends the required artifacts along with the secure ZTP yang model to the onboarding device.

4. **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 4: Server Validation



5. **Artifact Validation**:

   The router validates the artifact received from the ZTP server.

   a. The device extracts the `pinned-domain-cert` node, an X.509 certificate from the ownership voucher to verify the owner certificate.

   b. The device authenticates the owner certificate by performing the X.509 certificate path verification process on the trusted certificate.

   c. Finally, the device verifies whether the conveyed information artifact is signed by the validated owner certificate.

6. **Provision the device**:

   a. The device first processes the boot image information.

   b. Executes the pre-configuration script and then commits the initial configuration

   c. Execute the post configuration script.

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

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

Figure 5: End-to-end sequence of the Secure ZTP process

Disable Secure ZTP

Execute the following commands to disable the secure ZTP:

```
Router# request consent-token generate-challenge secure-ztp auth-timeout 15
Router# request consent-token accept-challenge secure-ztp
```