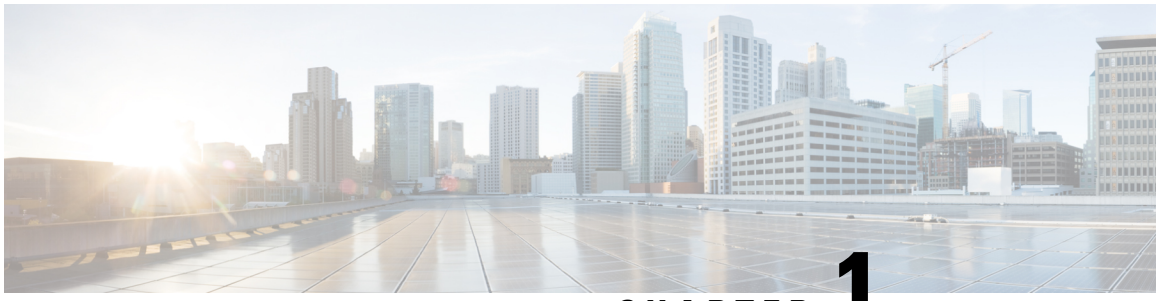
Cisco Embedded Service 9300 Series Switches Configuration Guide

First Published: 2020-10-12

Last Modified: 2023-01-04

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

Cisco Embedded Service 9300 Series Switches Software Configuration Overview

This section contains the following:

General Description

The Cisco ESS 9300 is a Small Form Factor (SFF) embedded Ethernet switch card. The compact design simplifies integration and offers system integrators the ability to use the Cisco ESS 9300 in a wide variety of applications. The Cisco ESS 9300 consists of one switch card. The ESS-9300-10X-E board supports up to 10 ports of 10 GE fiber, as well as the gig0/0 management port. The management port is active in ROMMON mode (IOS-XE not running). Thermal power is 35 Watts.

Important There are no cooling plates sold with the switch. It is up to the integrator to design a thermal solution

Some of the key hardware features are:

- Small form factor
- 10 Optical 10G
- Software: IOS-XE, Network Essentials and Network Advantage
- Industrial temperature: -40°C to +85°C
- ARM Quad-Core A53
- 4GB DDR4 DRAM memory capacity with ECC
- About 2.5GB usable eMMC flash

The port mapping for the hardware device to the IOS-XE device is as follows:

Table 1: Device Port Mapping

Device Port	IOS-XE Device
Management Port	GigabitEthernet0/0
Gigabit Ethernet Ports 1-10	TenGigabitEthernet1/1 - 1/10
USB	usbflash0:
mSATA	msata:

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

SD Support

There is one Cisco SD card that has been tested and is recommended, the SD-IE-4GB. If the end user or system integrator chooses to use a 3rd party device, it may work for their application and to their satisfaction. However the end user or system integrator is solely responsible for testing and ensuring proper operation.

The message that displays when a different SD card is installed is:

```
WARNING: Non-IT SD flash detected. Use of this card during normal operation can impact and
severely degrade performance of the system. Please use supported SD flash cards only.
```

You can find Cisco's policy on Third Party Components here:

https://www.cisco.com/c/en/us/products/warranties/warranty-doc-c99-740959.html#_Toc3320258

SFP Support

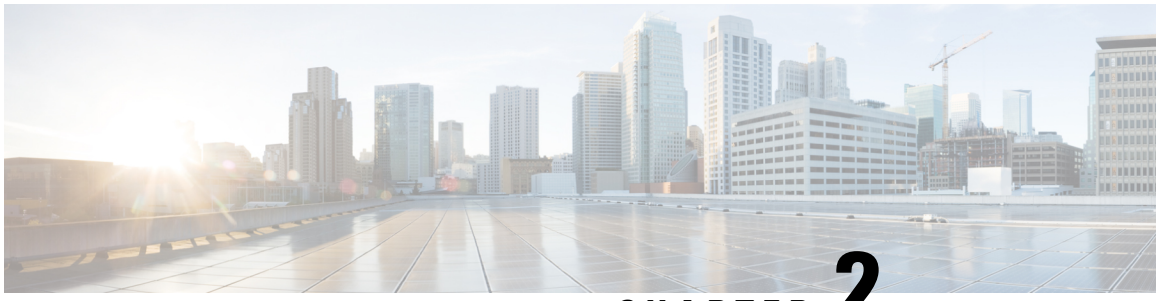
Both 100BASE-X and 1000BASE-X SFP transceivers are supported. Supported SFP and SFP+ Modules, on page 3 lists the specific SFP transceivers and their characteristics.

Note LRM optics are not supported since the SFP is direct driven from the Cisco ASIC.

Supported SFP and SFP+ Modules

Table 2: Supported Modules

Part Number	Specification	SFP Type	Max Distance	Cable Type	Temp Range	DOM Support
GLC-SX-MM-RGD=	1000BASE-SX	GE	550m	MMF	IND	Yes
GLC-LX-SM-RGD=	1000BASE-LX/LH	GE	550m/10km	MMF/SMF	IND	Yes
GLC-SX-MMD=	1000BASE-SX	GE	550m	MMF	EXT	Yes
GLC-LH-SMD=	1000BASE-LX/LH	GE	550m/10km	MMF/SMF	EXT	Yes
GLC-BX-D=	1000BASE-BX10	GE	10km	SMF	COM	Yes
GLC-BX-U=	1000BASE-BX10	GE	10km	SMF	COM	Yes
CWDM-SFP-xxxx= (8 freq)	CWDM 1000BASE-X	GE		SMF	COM	Yes
DWDM-SFP-xxxx= (40 freq)	DWDM 1000BASE-X	GE		SMF	COM	Yes
SFP-GE-S=	1000BASE-SX	GE	550m	MMF	EXT	Yes
SFP-GE-L=	1000BASE-LX/LH	GE	550m/10km	MMF/SMF	EXT	Yes
GLC-SX-MM=	1000BASE-SX	GE	550m	MMF	COM	No
GLC-LH-SM=	1000BASE-LX/LH	GE	550m/10km	MMF/SMF	COM	No
GLC-TE=	1000BASE-T	GE	100m	Copper	EXT	N/A
GLC-T=	1000BASE-T	GE	100m	Copper	COM	N/A
SFP-10G-BXD-I=	10GBASE-BX10	10GE	10km	SMF	IND	Yes
SFP-10G-BXU-I=	10GBASE-BX10	10GE	10km	SMF	IND	Yes
SFP-10G-SR-X=	10GBASE-SR	10GE	400m	MMF	EXT	Yes
SFP-10G-LR-X=	10GBASE-LR	10GE	10km	SMF	EXT	Yes
SFP-10G-SR=	10GBASE-SR	10GE	400m	MMF	COM	Yes
SFP-10G-LR=	10GBASE-LR	10GE	10km	SMF	COM	yes
GLC-T-RGD=	1000BASE-T	GE	100m	Copper	IND	N/A
SFP-H10GB-CUxM=	10G Passive Twinax	10GE	1m/3m/5m	Twinax	COM	N/A
SFP-H10GB-ACUxM=	10G Active Twinax	10GE	7m/10m	Twinax	COM	N/A



CHAPTER 2

Installation and Boot

This section contains the following:

Configuring the Switch with the CLI-Based Setup Program

This section provides a command-line interface (CLI)-based setup procedure for a switch. You must be connected to the switch through the console port to use the CLI. The ESS9300 auto detects whether the console port is RJ-45 or USB.

If using an RJ-45 console connection, configure with these parameters:

- 9600 baud
- 8 data bits
- 1 stop bit
- No parity
- None (flow control)

If you are connecting the switch USB-mini console port to a Windows-based PC for the first time, install a USB driver. If your laptop or PC warns you that you do not have the proper drivers to communicate with the router, you can obtain them from your computers manufacturer, or go here: https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers

Start the terminal-emulation program on the PC or the terminal. The program, frequently a PC application such as HyperTerminal or ProcommPlus, makes communication possible between the switch and your PC or terminal.

Connect power to the device. The PC or terminal displays the bootloader sequence. Press **Enter** to display the setup prompt.

Entering the Initial Configuration Information

To set up the switch, you need to complete the setup program, which runs automatically after the switch is powered on. You must assign an IP address and other configuration information necessary for the switch to communicate with the local routers and the Internet. This information is also required if you plan to use Web UI to configure and manage the switch.

IP Settings

You need this information from your network administrator before you complete the setup program:

- Switch IP address
- Subnet mask (IP netmask)
- Default gateway (router)
- Enable secret password
- Enable password

Completing the Setup Program

To complete the setup program and to create an initial configuration for the switch:

1. Enter **Yes** at these two prompts:

```
Would you like to enter the initial configuration dialog? [yes/no]: yes
At any point you may enter a question mark '?' for help.
Use ctrl-c to abort configuration dialog at any prompt.
Default settings are in square brackets '[]'.
Basic management setup configures only enough connectivity
for management of the system, extended setup will ask you
to configure each interface on the system.
Would you like to enter basic management setup? [yes/no]: yes
```

2. Enter a hostname for the switch, and press **Return**.

```
Enter host name [Switch]: host_name
```

3. Enter an enable secret password, and press **Return**.

   The password can be from 1 to 25 alphanumeric characters, can start with a number, is case sensitive, allows spaces, but ignores leading spaces. The secret password is encrypted, and the enable password is in plain text.

```
Enter enable secret: secret_password
```

4. Enter an enable password, and press **Return**.

```
Enter enable password: enable_password
```

5. Enter a virtual terminal (Telnet) password, and press **Return**.

   The password can be from 1 to 25 alphanumeric characters, is case sensitive, allows spaces, but ignores leading spaces.

```
Enter virtual terminal password: terminal-password
```

6. (Optional) Configure Simple Network Management Protocol (SNMP) by responding to the prompts. You can also configure SNMP later through the CLI, Device Manager, or the Cisco Network Assistant application. To configure SNMP later, enter **no**.

```
Configure SNMP Network Management? [no]: no
```

7. Enter the interface name (physical interface or VLAN name) of the interface that connects to the management network, and press **Return**. For this release, always use **vlan1** as that interface.

```
Current interface summary
Any interface listed with OK? value "NO" does not have a valid configuration
Interface IP-Address OK? Method Status Protocol
Vlan1 unassigned NO unset up down
TenGigabitEthernet1/1 unassigned YES unset down down
TenGigabitEthernet1/2 unassigned YES unset down down
TenGigabitEthernet1/3 unassigned YES unset down down
TenGigabitEthernet1/4 unassigned YES unset down down
TenGigabitEthernet1/5 unassigned YES unset down down
TenGigabitEthernet1/6 unassigned YES unset down down
TenGigabitEthernet1/7 unassigned YES unset down down
TenGigabitEthernet1/8 unassigned YES unset down down
TenGigabitEthernet1/9 unassigned YES unset down down
TenGigabitEthernet1/10 unassigned YES unset down down
Enter interface name used to connect to the
management network from the above interface summary: vlan1
Enter interface name used to connect to the
management network from the above interface summary: vlan1
```

8. Configure the interface by entering the switch IP address and subnet mask and pressing Return. The IP address and subnet masks shown here are examples.

```
Configuring interface Vlan1:
Configure IP on this interface? [yes]:
IP address for this interface: 10.1.1.2
Subnet mask for this interface [255.255.255.0] :
Class A network is 10.0.0.0, 8 subnet bits; mask is /24
```

9. This summary appears:

```
The following configuration command script was created:
hostname ESS9300
enable secret 9 $9$rkqtjJhIkZyANU$Ib4nfuxrpHBi.lixF.0Ir94k9XWYsW3nyF7Glmc6lkc
enable password cisco
line vty 0 15
password cisco
no snmp-server
!!
interface Vlan1
no shutdown
ip address 10.1.1.2 255.255.255.0
!
interface TenGigabitEthernet1/1
!
interface TenGigabitEthernet1/2
!
interface TenGigabitEthernet1/3
!
interface TenGigabitEthernet1/4
!
interface TenGigabitEthernet1/5
!
```

```
interface TenGigabitEthernet1/6
!
interface TenGigabitEthernet1/7
!
interface TenGigabitEthernet1/8
!
interface TenGigabitEthernet1/9
!
interface TenGigabitEthernet1/10
!
end
```

After you complete the setup program, the switch can run the default configuration that you created. If you want to change this configuration or want to perform other management tasks, use the CLI or WEBUI.

To use the CLI, enter commands at the Switch> prompt through the console port by using a terminal emulation program or through the network by using Telnet.

Upgrading the Switch Software

This section covers the various aspects of upgrading or downgrading the device software.

Finding the Software Version

The package files for the Cisco IOS XE software can be found on the system board flash device flash (flash:) or external SDFlash (sdflash:).

You can use the **show version** privileged EXEC command to see the software version that is running on your switch.

Note Although the **show version** output always shows the software image running on the switch, the model name shown at the end of this display is the factory configuration and does not change if you upgrade the software license.

You can also use the **dir** *filesystem:* privileged EXEC command to see the directory names of other software images that you might have stored in flash memory.

Software Images

The switch runs on Cisco IOS-XE with an initial release of 17.4.1. The switch uses a universal image type named ie9k_iosxe.<release>.SPA.bin.

Automatic Boot Loader Upgrade

When you upgrade from the existing release on your switch to a later or newer release for the first time, the boot loader may be automatically upgraded, based on the hardware version of the switch. If the boot loader is automatically upgraded, it will take effect on the next reload.

For subsequent Cisco IOS XE releases, if there is a new bootloader in that release, it may be automatically upgraded based on the hardware version of the switch when you boot up your switch with the new image for the first time.

Caution Do not power cycle your switch during the upgrade.

Scenario	Automatic Boot Loader Response

If you boot Cisco IOS XE the first time	Boot loader may be upgraded to version "7.1.5" for ESS-9300.
	`Checking Bootloader upgrade...`
	`…`
	`Bootloader upgrade successful`

Bundle Mode Upgrade

To upgrade the Cisco IOS XE software when the switch is running in bundle mode, follow these steps:

Step 1 Download the bundle file to local storage media.

Step 2 Configure the **boot system** global configuration command to point to the bundle file.

Step 3 Reload the switch.

Example

Upgrading Cisco IOS XE Software Bundle Mode

This example shows the steps to upgrade the Cisco IOS XE software on a switch that is running in bundle mode. It shows using the **copy** command to copy the bundle file to flash:, configuring the boot system variable to point to the bundle file, saving a copy of the running configuration, and finally, reloading the switch.

```
Switch#copy scp: sdflash:
Address or name of remote host [10.106.224.22]?
Source username [xxxxx]?
Source filename []? $2/binos/linkfarm/iso1-petra/ie9k-universalk9.17.04.01.SPA.bin
Destination filename [ie9k-universalk9.17.04.01.SPA.bin]?
This is a Cisco managed device to be used only for authorized purposes.
Your use is monitored for security, asset protection, and policy compliance.


Password:
 Sending file modes: C0644 344345038 ie9k-universalk9.17.04.01.SPA.bin
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
344345038 bytes copied in 637.684 secs (539993 bytes/sec)

Switch#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#no boot system
Switch(config)#boot system sdflash:ie9k-universalk9.17.04.01.SPA.bin
Switch(config)#end
```

```
Switch#write memory
*May 27 14:49:55.121: %SYS-5-CONFIG_I: Configured from console by console
Building configuration...
[OK]
Switch#s
*May 27 14:50:01.341: %SYS-6-PRIVCFG_ENCRYPT_SUCCESS: Successfully encrypted private config
 file

Switch#show boot
Current Boot Variables:
BOOT variable = sdflash:ie9k-universalk9.17.04.01.SPA.bin;

Boot Variables on next reload:
BOOT variable = sdflash:ie9k-universalk9.17.04.01.SPA.bin;
Config file = flash:/nvram_config
ENABLE_FLASH_PRIMARY_BOOT = no
MANUAL_BOOT variable = no
ENABLE_BREAK variable = yes

Switch#reload
Proceed with reload? [confirm]

*May 27 14:50:08.989: %SYS-5-RELOAD: Reload requested by console. Reload Reason: Reload
Command.
watchdog: watchdog0: watchdog did not stop!
reboot: Restarting system
```

Software Installation Commands

Summary of Software Installation Commands	
To install and activate the specified file, and to commit changes to be persistent across reloads—**install add file** *filename* [**activate commit**]	
add file tftp: *filename*	Copies the install file package from a remote location to the device and performs a compatibility check for the platform and image versions.
activate [**auto-abort-timer**]	Activates the file, and reloads the device. The **auto-abort-timer** keyword automatically rolls back image activation.
commit	Makes changes persistent over reloads.
remove	Deletes all unused and inactive software installation files.

Licensing

This section provides information about the licensing packages for features available on Cisco ESS9300 series switches.

License Levels

The initial release of the ESS9300 has only the Network Essentials license.

Note Network Essentials license is the default license. It is permanent. A connection to the Smart Licensing server is not required if the switch will be deployed with a Network Essentials license.

Note Entering the command **license smart reservation** after the initial configuration will prevent an erroneous message "Smart Licensing Status: UNREGISTERED/EVAL MODE" from appearing on your device.

Boot from the USB

The switch can be booted from configuration files located on the pluggable USB. Customized startup configuration files can be booted from IOS or from ROMMON.

Booting from IOS

The following configuration steps need to be taken in order to boot from the USB.

To display the boot options:

```
switch(config)#boot config ?
  bootflash:  URL of the config file
  flash:      URL of the config file
  msata:      URL of the config file
  nvram:      URL of the config file
  usbflash0:  URL of the config file
  webui:      URL of the config file
```

The syntax for the boot command is:

boot config usbflash0:<*file name*>

For example:

```
switch(config)#boot config usbflash0:startup-config
switch(config)#
switch#write memory
Building configuration...
[OK]
*Feb 10 10:20:11.990: %SYS-2-PRIVCFG_ENCRYPT: Successfully encrypted private config file
```

The environment variable CONFIG_FILE in the following example confirms that the startup-config is set to boot from usbflash0.

```
switch#show boot
BOOT variable =
CONFIG_FILE variable = usbflash0:startup-config
BOOTLDR variable does not exist
Configuration register is 0x1820
Standby not ready to show bootvar
```

Booting from ROMMON

The following configuration steps need to be taken in order to boot from the USB.

From the ROMMON prompt, execute **set CONFIG_FILE=usbflash0:** <*filename*>

For example:

```
rommon 2 > set CONFIG_FILE=usbflash0:my_startupcfg
rommon 3 > sync
rommon 4 > set
PS1=rommon ! >
MCU_UPGRADE=SKIP
THRPUT=
LICENSE_BOOT_LEVEL=
RET_2_RTS=
MCP_STARTUP_TRACEFLAGS=00000000:00000000
BSI=0
RANDOM_NUM=1275114933
BOOT=flash:Jun5_1.SSA,12
RET_2_RCALTS=951454376
CONFIG_FILE=usbflash0:my_startupcfg
```

Continue booting the IOS image as usual from the ROMMON prompt.

Booting from the USB Feature Summary

- Once the CONFIG_FILE is set to a non-default value, the **nvram:startup-config** command is aliased to this new location.
- Any change made to the config file in usbflash will be reflected in nvram:startup-config as well.
- The EXEC command **erase nvram:startup-config** erases the contents of NVRAM, and deletes the file referenced by CONFIG_FILE variable.
- If the USB is unplugged after setting the **boot config usbflash0:** <*filename*> variable, then the day 0 default configuration will take effect.
- When the configuration is saved using the **copy system:running-config nvram:startup-config** command, the device saves a complete version of the configuration file to the location specified by the CONFIG_FILE environment variable, and a distilled version to NVRAM. A distilled version is one that does not contain access list information.

Emergency Recovery Installation

Note There is different terminology used when referring to the reset button depending on the product. The IE3x00 switches call this the Express Setup switch. Other products may refer to this as the Factory Default Switch. In either case, the functionality is the same.

If the other recovery methods fail, the switch has a trap door method that you can use in order to recover the system. You must have a terminal that is connected to port Gi1/3 of the switch that runs a TFTP server. Download a valid image file from CCO and store it in the root of the TFTP server.

It is likely that the switch is stuck at the **switch**: prompt. However, if you are in a boot loop, you can use the Express Setup switch on the front of the switch in order to break the cycle: hold the button for approximately <TBD> seconds, and the switch breaks the cycle and stops at the **switch**: prompt.

Complete these steps in order to perform an emergency recovery:

Step 1: Boot the emergency install image.

```
switch: switch: boot emgy0:<image-name>.SPA.bin
Booting golden bootloader...
Initializing disk drivers...
Initializing file systems...
**********************************************************
* Rom Monitor for ESS3300                                *
* Copyright (c) 2017-2018 by Cisco Systems, Inc.         *
* All rights reserved.                                   *
**********************************************************
* Version:  1.1.1
* Compiled: Sun 01-Jul-18 22:17 [RELEASE SOFTWARE]
* Boot Partition: qspi-golden-bootloader
* Reset Reason: Soft Reset
Loading "emgy0:ess3x00-universalk9.17.04.01.SPA.bin" to memory...
Verifying image "emgy0:ess3x00-universalk9.17.04.01.SPA.bin"...
Image passed digital signature verification
Checking for Bootloader upgrade...
Bootloader upgrade not required
SUP PL (profile: 1) configuration done successfully
<...>
Press RETURN to get started!
Switch>
```

Step 2: Configure an IP address on the switch. Additional details on IP configuration can be found here

```
switch(config-if)# ip address <ip-address> <subnet-mask>
```

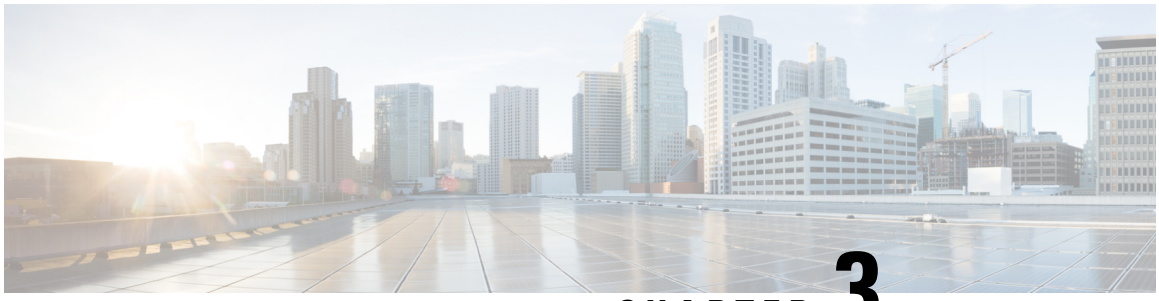
Step 3: Ping the terminal that contains the TFTP server in order to test the connectivity:

```
switch> ping 192.168.2.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echoes to 192.168.2.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

Step 4: Copy the image via tftp

```
switch> copy tftp: //location/directory/<bundle_name> flash:
<...>
```

Step 5: Restart the system.



CHAPTER 3

Introduction to Day 0 WebUI Configuration

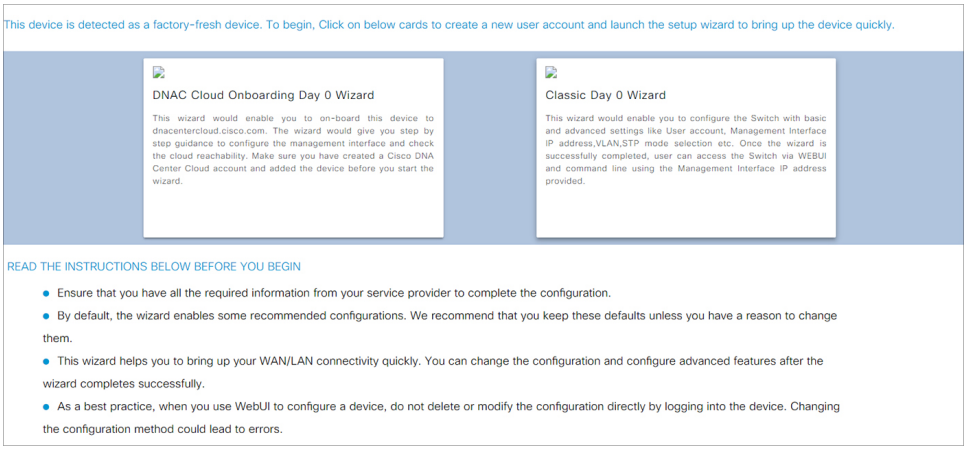
This chapter contains the following sections:

Classic Day 0 Wizard

After you complete the hardware installation, you need to setup the switch with configuration required to enable traffic to pass through the network. On your first day with your new device, you can perform a number of tasks to ensure that your device is online, reachable and easily configured.

The Web User Interface (Web UI) is an embedded GUI-based device-management tool that provides the ability to provision the device, to simplify device deployment and manageability, and to enhance the user experience. You can use WebUI to build configurations, monitor, and troubleshoot the device without having CLI expertise.

Figure 1: WebUI Day 0 Wizard



Use this wizard to configure the device with basic and advanced settings. Once complete, you can access the device through the WebUI using the management interface IP address.

Connecting to the Switch

Before you begin

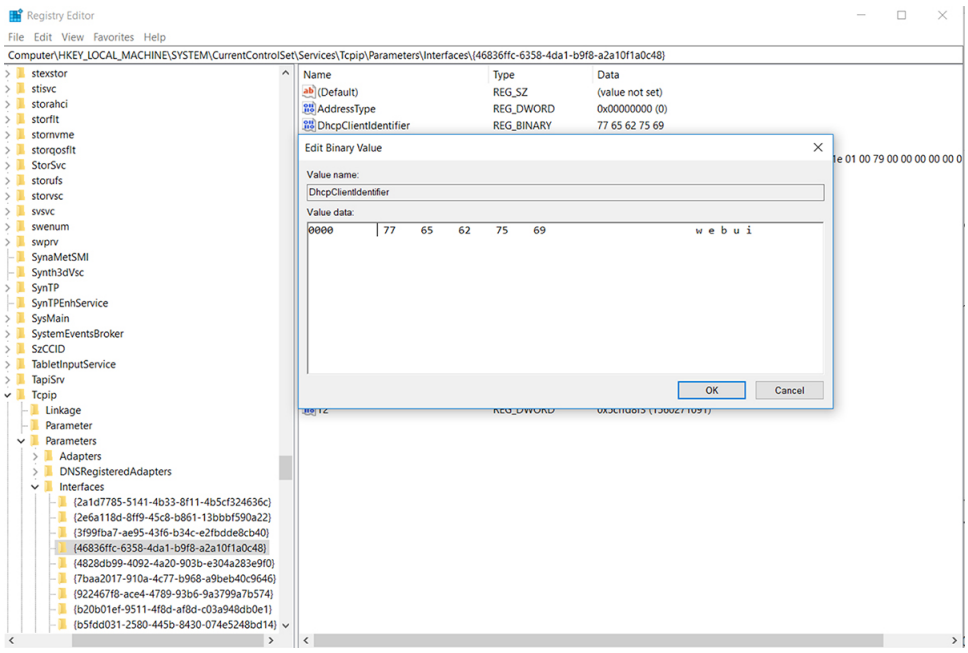
Set up the DHCP Client Identifier on the client to get the IP address from the switch, and to be able to authenticate with Day 0 login credentials.

Setting up the DHCP Client Identifier on the client for Windows

1. Type **`regedit`** in the Windows search box on the taskbar and press *enter*.
2. If prompted by User Account Control, click **Yes** to open the Registry Editor.
3. Navigate to

   Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces and locate the **Ethernet Interface** Global Unique Identifier (GUID).
4. Add a new REG_BINARY **DhcpClientIdentifier** with Data **77 65 62 75 69** for **`webui`**. You need to manually type in the value.

Figure 2: Setting up DHCP Client Identifier on Windows

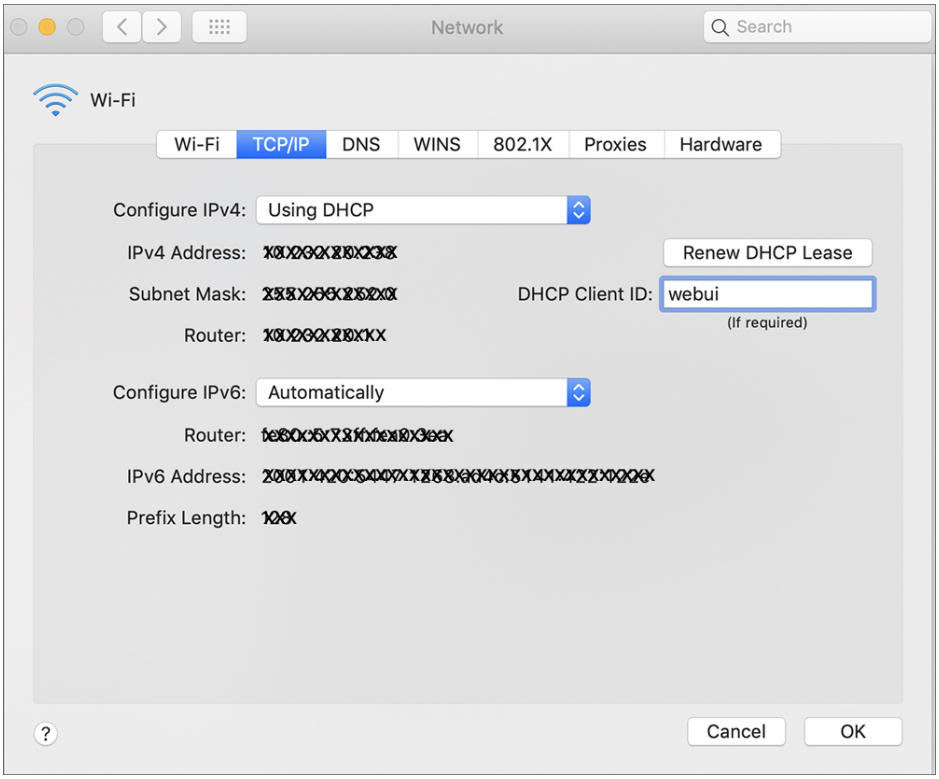


5. Restart the PC for the configuration to take effect.

Setting up the DHCP Client Identifier on the client for MAC

1. Go to **System Preferences >Network >Advanced >TCP >DHCP Client ID:** and enter **`webui`**.

Figure 3: Setting up DHCP Client Identifier on MAC



2. Click **OK** to save the changes.

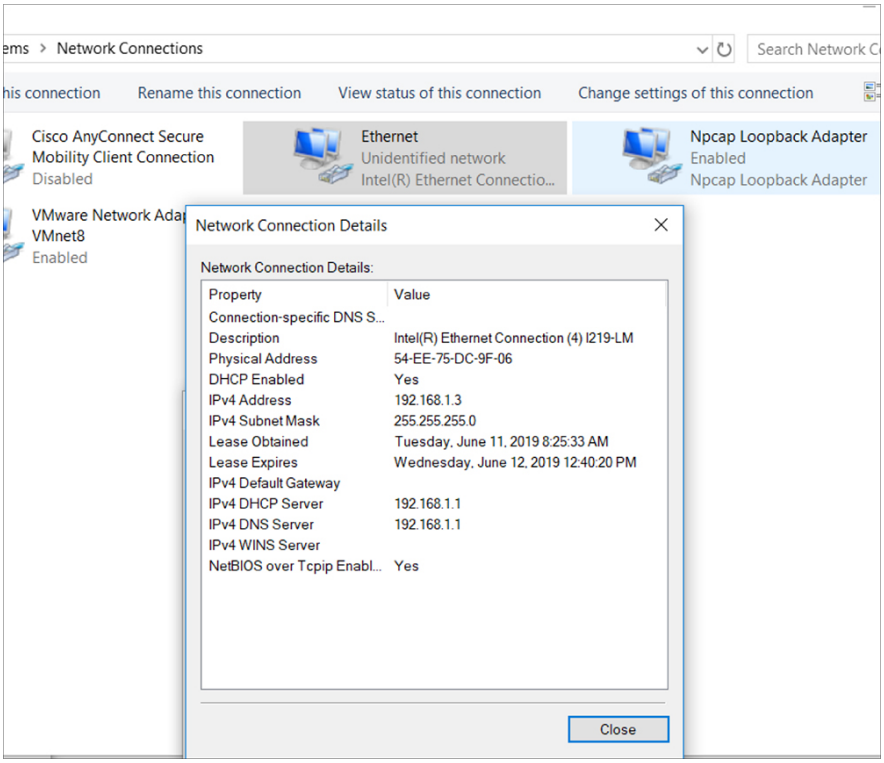
The bootup script runs the configuration wizard, which prompts you for basic configuration input: (**Would you like to enter the initial configuration dialog? [yes/no]:**). To configure Day 0 settings using the web UI, do not enter a response. Perform the following tasks instead:

Step 1 Make sure that no devices are connected to the switch.

Step 2 Connect one end of an ethernet cable to one of the downlink (non-management) ports on the active supervisor and the other end of the ethernet cable to the host (PC/MAC).

Step 3 Set up your PC/MAC as a DHCP client, to obtain the IP address of the switch automatically. You should get an IP address within the 192.168.1.x/24 range.

Figure 4: Obtaining the IP Address



It may take up to three mins. You must complete the Day 0 setup through the web UI before using the device terminal.

Step 4 Launch a web browser on the PC and enter the device IP address (**`https://192.168.1.1`**) in the address bar.

Step 5 Enter the Day 0 **username `webui`** and **password `cisco`**.

What to do next

Create a user account.

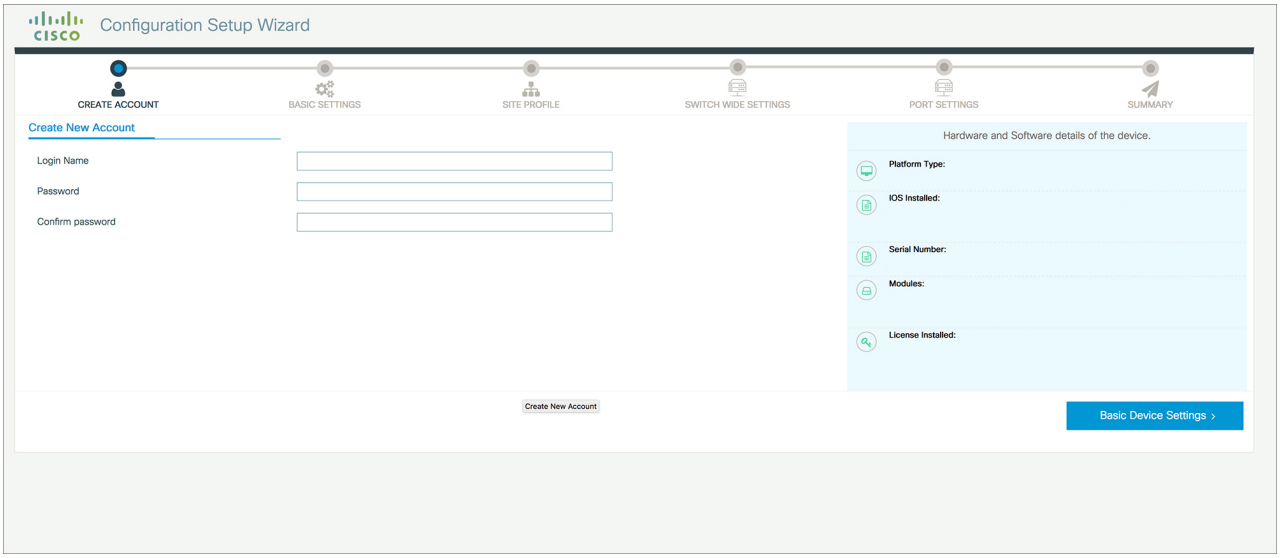
Creating User Accounts

Setting a username and password is the first task you will perform on your device. Typically, as a network administrator, you will want to control access to your device and prevent unauthorized users from seeing your network configuration or manipulating your settings.

Step 1 Log on using the default username and password provided with the device.

Step 2 Set a password of up to 25 alphanumeric characters. The username password combination you set gives you privilege 15 access. The string cannot start with a number, is case sensitive, and allows spaces but ignores leading spaces.

Figure 5: Create Account



Choosing Setup Options

Select **Wired Network** to configure your device based on a site profile, and continue to configure switch wide settings. Otherwise, continue to the next step and configure only basic settings for your device.

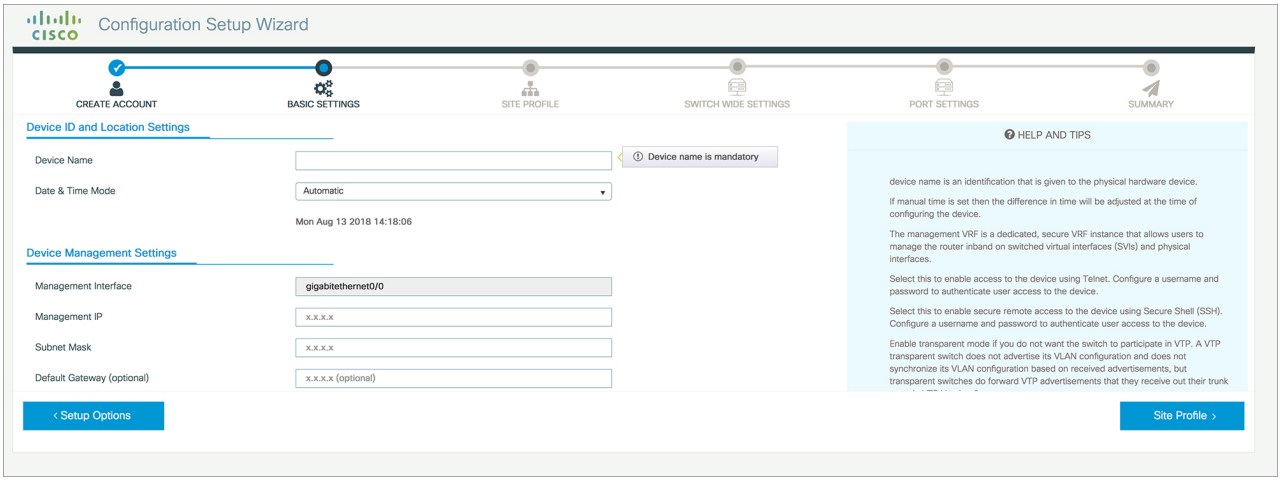
Configuring Basic Device Settings

On the **Basic Device Settings** page configure the following information:

Step 1 In the **Device ID and Location Settings** section, type a unique name to identify your device in the network.

Step 2 Choose the date and time settings for your device. To synchronize your device with a valid outside timing mechanism, such as an NTP clock source, choose Automatic, or choose Manual to set it yourself.

Figure 6: Basic Settings - Device ID and Location Settings



Step 3 In the **Device Management Settings** section, assign an **`IP address`** to the management interface. Ensure that the IP address you assign is part of the subnet mask you enter.

Step 4 Optionally, enter an **`IP address`** to specify the default gateway.

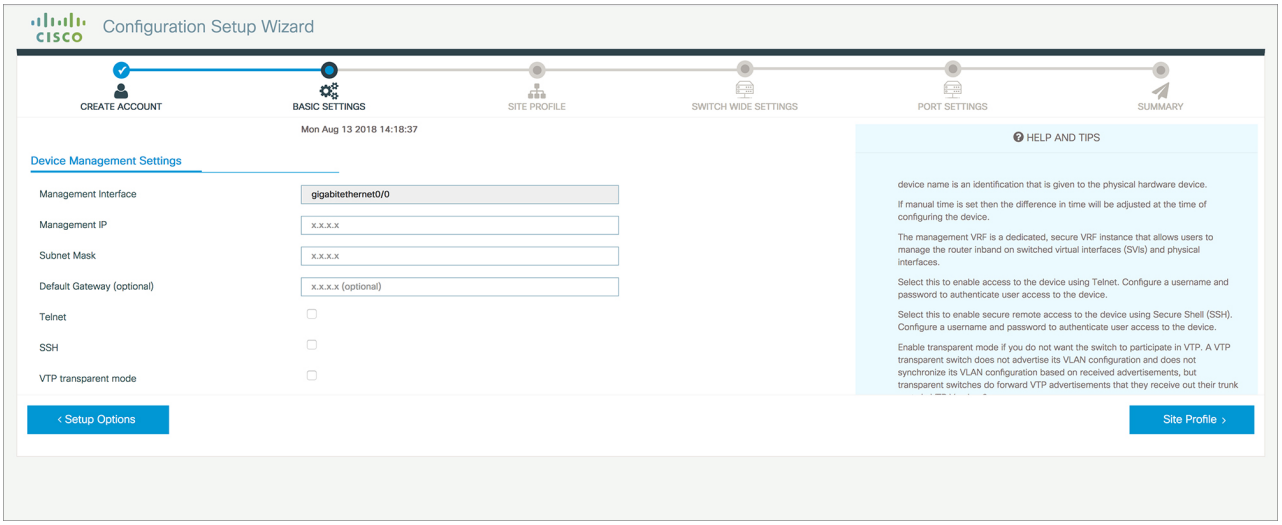
Step 5 To enable access to the device using telnet, check the **Telnet** check box.

Step 6 To enable secure remote access to the device using Secure Shell (SSH), check the **SSH** check box.

Step 7 Check the **VTP transparent mode** check box to disable the device from participating in VTP.

If you did not select **Wired Network**, in the earlier step, continue to the next screen to verify your configuration on the **Day 0 Config Summary** screen, and click **Finish**. To automatically configure your device based on a site profile, click **Setup Options**, and select **Wired Network**.

Figure 7: Basic Settings - Device Management Settings



Configuring Your Device Based on a Site Profile

To ease your configuration tasks and save time, choose a site profile based on where your device may be installed and managed in your network. Based on the site profile you choose, your device is automatically configured according to Cisco best practices. You can easily modify this default configuration, from the corresponding detailed configuration screens.

Choosing a site profile as part of Quick Setup allows you to configure your device based on the business needs of your enterprise. For example, you could use your device as an access switch, to connect client nodes and endpoints on your network, or as a distribution switch, to route packets between subnets and VLANs.

Table 3: Default Configuration Loaded with Each Site Profile (Access Switches)

Setting	Single Access Switch (Single Uplink)	Single Access Switch (Single Port Channel Uplink)	Single Access Switch (Redundant Port Channel Uplink)
Hostname	The hostname or device name you provided as part of Quick Setup	The hostname or device name you provided as part of Quick Setup	The hostname or device name you provided as part of Quick Setup
Spanning Tree Mode	RPVST+	RPVST+	RPVST+
VTP	Mode Transparent	Mode Transparent	Mode Transparent
UDLD	Enabled	Enabled	Enabled
Error Disable Recovery	Recovery mode set to Auto	Recovery mode set to Auto	Recovery mode set to Auto
Port Channel Load Balance	Source Destination IP	Source Destination IP	Source Destination IP
SSH	Version 2	Version 2	Version 2
SCP	Enabled	Enabled	Enabled
VTY Access to Switch	Enabled	Enabled	Enabled
Service Timestamp	Enabled	Enabled	Enabled
VLAN	The following VLANs are created: • Default VLAN • Data VLAN • Voice VLAN • Management VLAN	The following VLANs are created: • Default VLAN • Data VLAN • Voice VLAN • Management VLAN	The following VLANs are created: • Default VLAN • Data VLAN • Voice VLAN • Management VLAN

Setting	Single Access Switch (Single Uplink)	Single Access Switch (Single Port Channel Uplink)	Single Access Switch (Redundant Port Channel Uplink)
Management Interface	Layer 3 settings configured on the management port, based on Quick Setup	Layer 3 settings configured on the management port, based on Quick Setup	Layer 3 settings configured on the management port, based on Quick Setup
IPv6 Host Policy	IPv6 host policy created	IPv6 host policy created	IPv6 host policy created
QoS Policy for Downlink Ports	Auto QoS Policy for Access defined	Auto QoS Policy for Access defined	Auto QoS Policy for Access defined
QoS Policy for Uplink Ports	QoS Policy for Distribution created	QoS Policy for Distribution created	QoS Policy for Distribution created
Uplink Interfaces	Selected uplink interfaces configured as trunk ports, set to allow all VLANs	Selected ports configured as Port-channel in trunk mode, set to allow all VLANs.	Selected ports configured as Port-channel in trunk mode, set to allow all VLANs.
Downlink Interfaces	Downlink ports configured in Access mode	Downlink ports configured in Access mode	Downlink ports configured in Access mode
Port-channel	Not configured	Port-channel to distribution created	Port-channel to distribution created

Figure 8: Site Profile - Access Switches

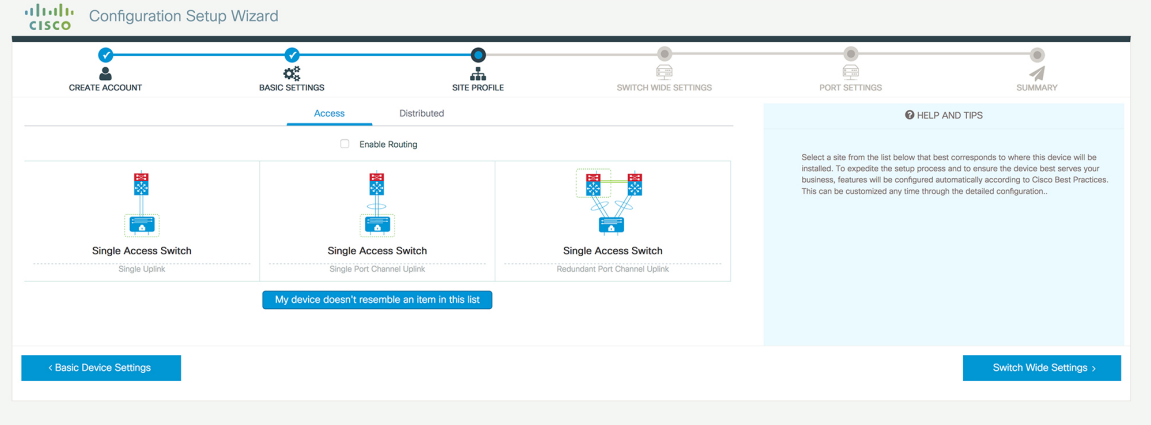


Figure 9: Site Profile - Access Switches (with Routed Access)

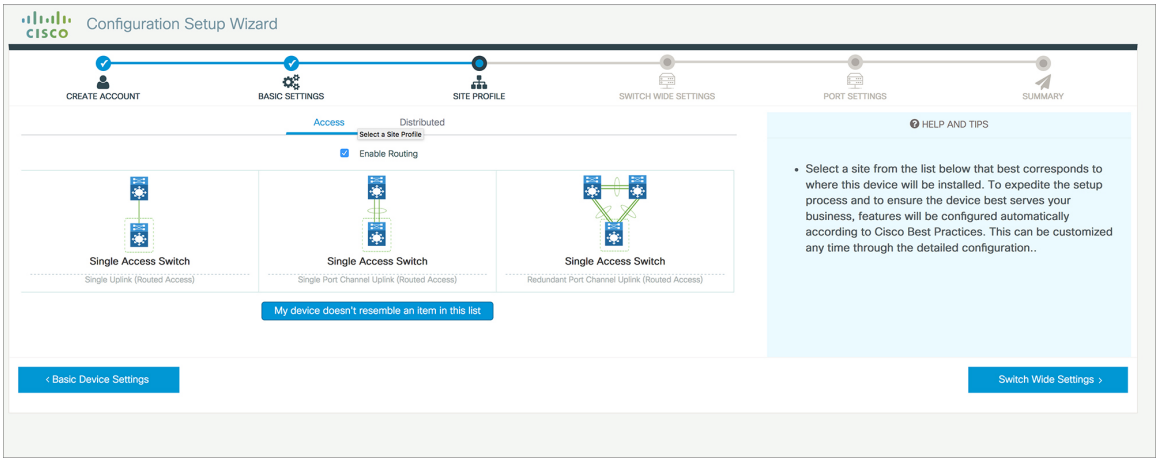


Table 4: Default Configuration Loaded with Each Site Profile (Distribution Switches)

Setting	Single Distribution Switch (Single Downlink)	Single Distribution Switch (Single Port Channel Downlink)	Redundant Distribution Switch (Port Channel Peer and Downlink)
Hostname	The hostname or device name you provided as part of Quick Setup	The hostname or device name you provided as part of Quick Setup	The hostname or device name you provided as part of Quick Setup
Spanning Tree Mode	RPVST+	RPVST+	RPVST+
VTP	Mode Transparent	Mode Transparent	Mode Transparent
UDLD	Enabled	Enabled	Enabled
Error Disable Recovery	Recovery mode set to Auto	Recovery mode set to Auto	Recovery mode set to Auto
SSH	Version 2	Version 2	Version 2
SCP	Enabled	Enabled	Enabled
VTY Access to Switch	Enabled	Enabled	Enabled
Service Timestamp	Enabled	Enabled	Enabled
VLAN	The following VLANs are created: • Default VLAN • Data VLAN • Voice VLAN • Management VLAN	The following VLANs are created: • Default VLAN • Data VLAN • Voice VLAN • Management VLAN	The following VLANs are created: • Default VLAN • Data VLAN • Voice VLAN • Management VLAN

Setting	Single Distribution Switch (Single Downlink)	Single Distribution Switch (Single Port Channel Downlink)	Redundant Distribution Switch (Port Channel Peer and Downlink)
Management Interface	Layer 3 settings configured on the management port, based on Quick Setup	Layer 3 settings configured on the management port, based on Quick Setup	Layer 3 settings configured on the management port, based on Quick Setup
QoS Policy	QoS Policy for Distribution defined	QoS Policy for Distribution defined	QoS Policy for Distribution defined
Uplink Interfaces	Selected uplink ports connect to other distribution or core switches	Selected uplink ports connect to other distribution or core switches	Selected uplink ports connect to other distribution or core switches
Downlink Interfaces	Downlink connections to access switches configured in Trunk mode	Downlink connections to access switches configured in Trunk mode	Downlink connections to access switches configured in Trunk mode
Port-channel	Port-channel to core created	Port-channel to core or access created	Port-channel to core or distribution created

Figure 10: Site Profile - Distribution Switches

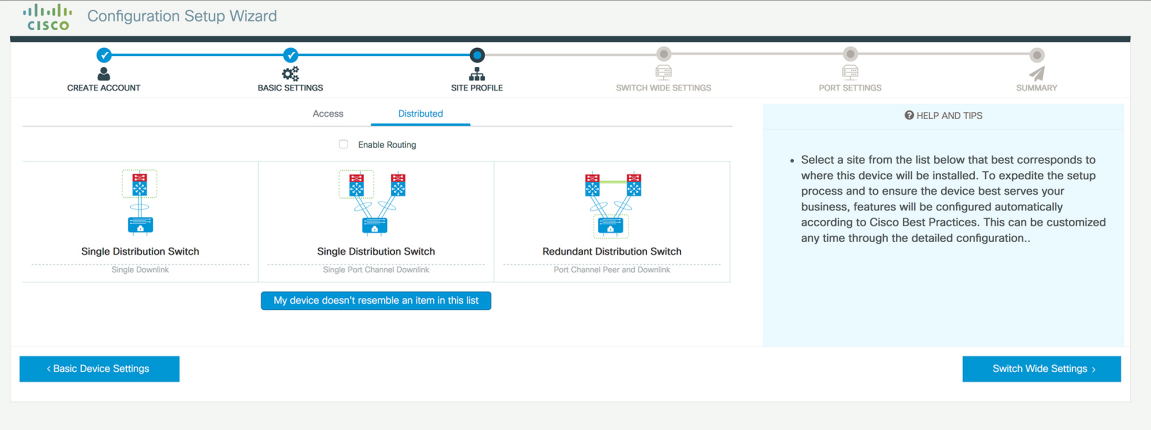


Figure 11: Site Profile - Distribution Switches (with Routed Access)

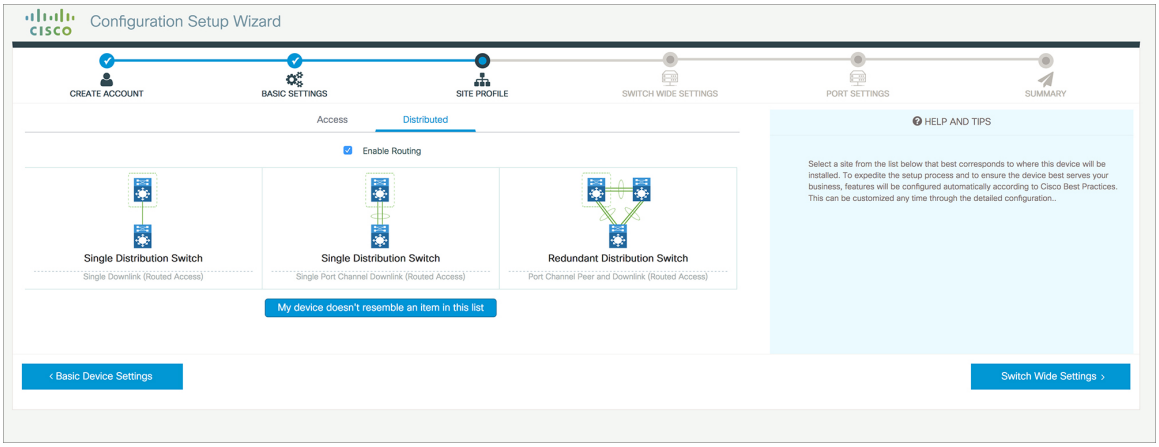
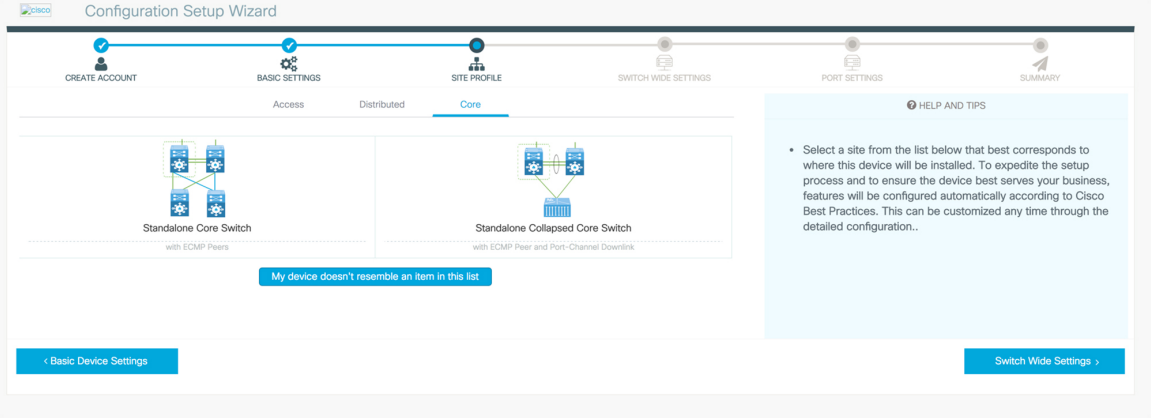


Table 5: Default Configuration Loaded with Each Site Profile (Core Switches)

Setting	Standalone Core Switch (with ECMP Peers)	Standalone Collapsed Core Switch (with ECMP Peer and Port Channel Downlink)
Hostname	The hostname or device name you provided as part of Quick Setup	The hostname or device name you provided as part of Quick Setup
UDLD	Enabled	Enabled
Error Disable Recovery	Recovery mode set to Auto	Recovery mode set to Auto
Port Channel Load Balance	Source Destination IP	Source Destination IP
SSH	Version 2	Version 2
SCP	Enabled	Enabled
VTY Access to Switch	Enabled	Enabled
Mitigate Address Spoofing	Unicast RPF (uRPF) in strict mode	Unicast RPF (uRPF) in strict mode
Service Timestamp	Enabled	Enabled
Management Interface	Layer 3 settings configured on the management port, based on Quick Setup	Layer 3 settings configured on the management port, based on Quick Setup
QoS Policy	QoS Policy for Distribution/Core defined	QoS Policy for Distribution/Core defined
Uplink Interfaces	Selected uplink ports connect to MAN/WAN device	Selected uplink ports connect to MAN/WAN device
Downlink Interfaces	Downlink connections to access switches	Downlink connections to distribution switches

Setting	Standalone Core Switch (with ECMP Peers)	Standalone Collapsed Core Switch (with ECMP Peer and Port Channel Downlink)
Cross-connect Interfaces	Selected ports connect to other core switches	Selected ports connect to other core switches

Figure 12: Site Profile - Core Switches



Configuring VLAN Settings

Step 1 In the **VLAN Configuration** section, you can configure both data and voice VLANs. Type a name for your data VLAN.

Step 2 To configure a data VLAN, ensure that the **Data VLAN** check box is checked, type a name for your VLAN, and assign a VLAN ID to it. If you are creating several VLANs, indicate only a VLAN range.

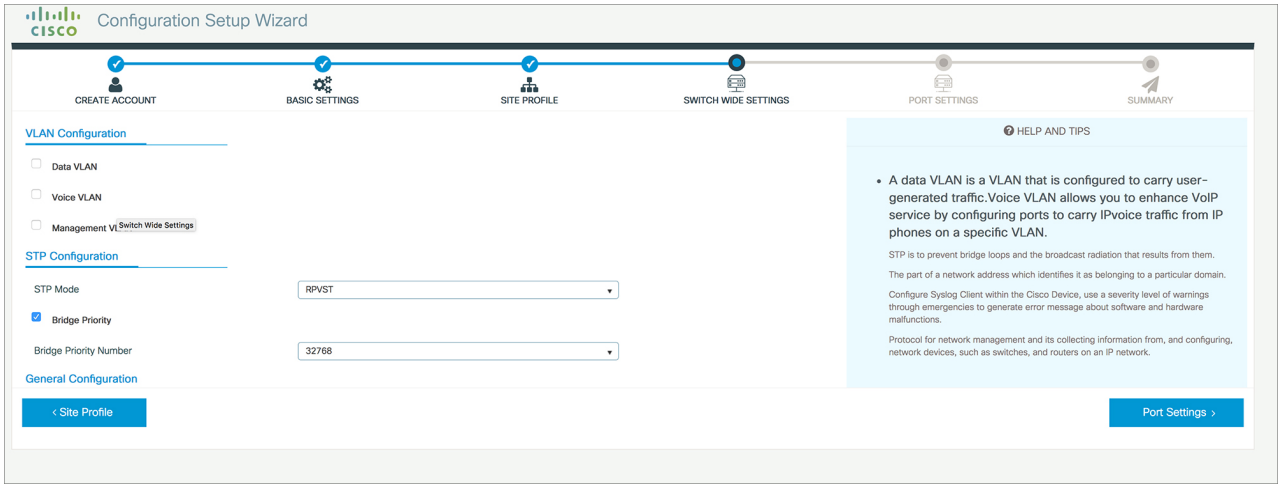
Step 3 To configure a voice VLAN, ensure that the **Voice VLAN** check box is checked, type a name for your VLAN, and assign a VLAN ID to it. If you are creating several VLANs, indicate a VLAN range.

Configure STP Settings

Step 1 RPVST is the default STP mode configured on your device. You can change it to PVST from the **STP Mode** drop-down list.

Step 2 To change a bridge priority number from the default value 32748, change **Bridge Priority** to Yes and choose a priority number from the drop-down list.

Figure 13: VLAN and STP Settings



Configuring DHCP, NTP, DNS and SNMP Settings

Step 1 In the **Domain Details** section, enter a domain name that the software uses to complete unqualified hostnames.

Step 2 Type an IP address to identify the DNS server. This server is used for name and address resolution on your device.

Step 3 In the **Server Details** section, type the IP address of the DNS server that you want to make available to DHCP clients.

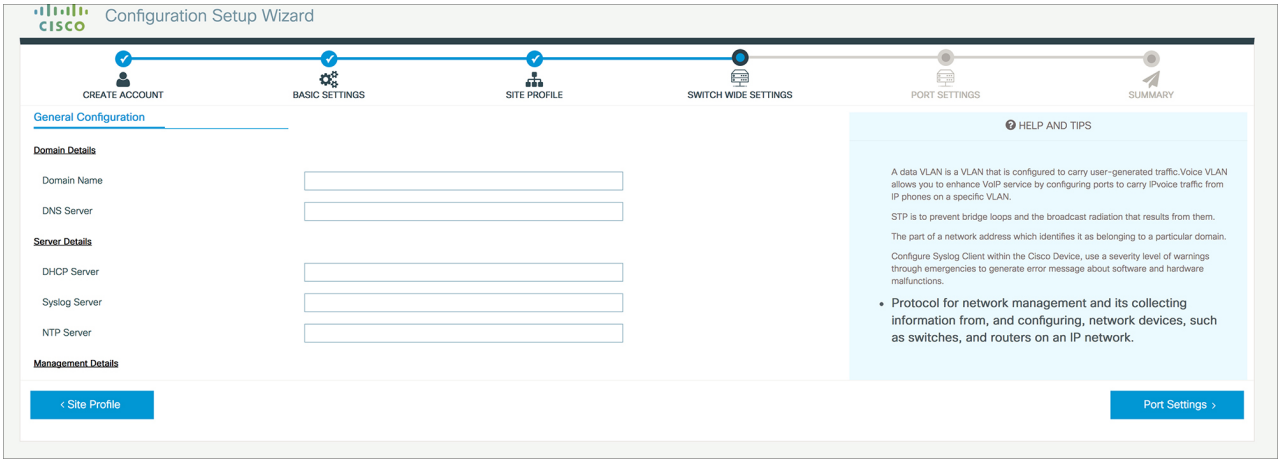
Step 4 In the **Syslog Server** field, type the IP address of the server to which you want to send syslog messages.

Step 5 To ensure that your device is configured with the right time, date and timezone, enter the IP address of the NTP server with which you want to synchronize the device time.

Step 6 In the **Management Details** section, type an IP address to identify the SNMP server. SNMPv1, SNMPv2, and SNMPv3 are supported on your device.

Step 7 Specify the **SNMP community** string to permit access to the SNMP protocol.

Figure 14: DHCP, NTP, DNS and SNMP Settings



What to do next

Configure port settings.

Configuring Port Settings

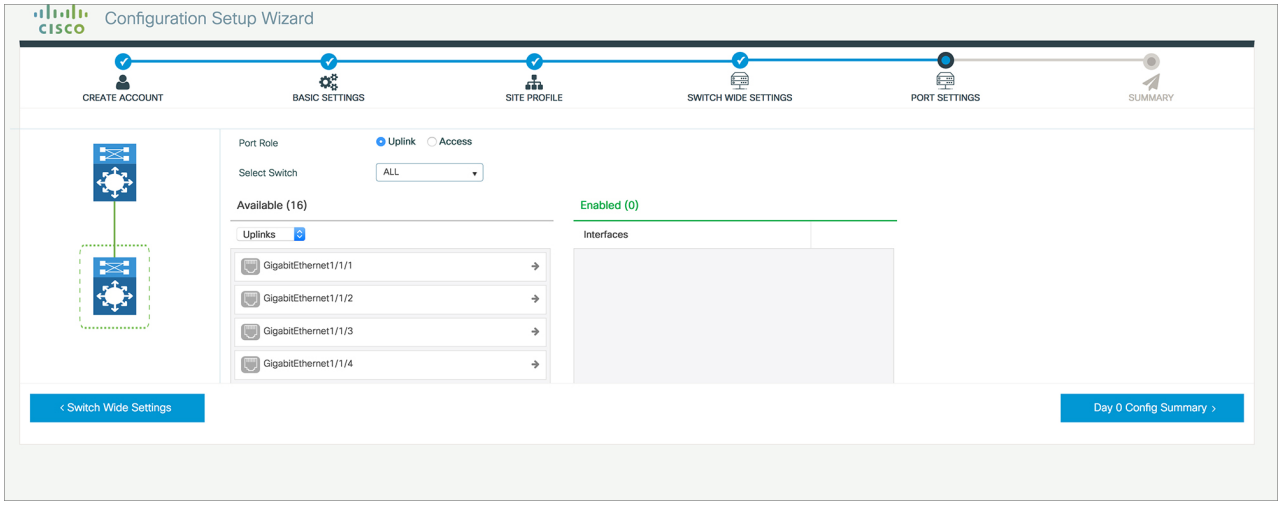
Step 1 Based on the site profile chosen in the earlier step which is displayed in the left-pane, select the **Port Role** from among the following options:

- Uplink – For connecting to devices towards the core of the network.
- Downlink – For connecting to devices further down in the network topology.
- Access – For connecting guest devices that are VLAN-unaware.

Step 2 Choose an option from the **Select Switch** drop-down list.

Step 3 Make selections from the **Available** list of interfaces based on how you want to enable them and move them to the **Enabled** list.

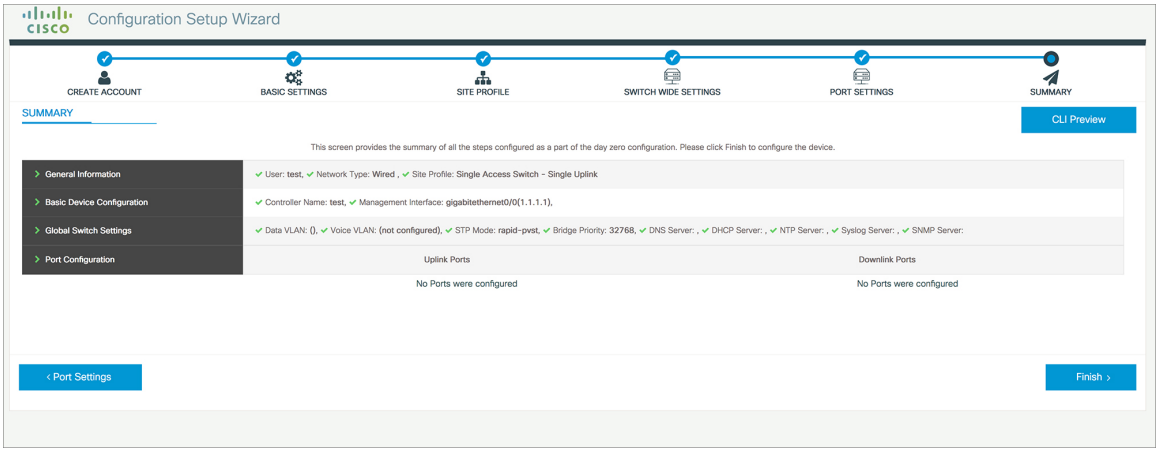
Figure 15: Port Settings



What to do next

- Click **Day 0 Config Summary** to verify your setup.
- Click **Finish**.

Figure 16: Day 0 Config Summary



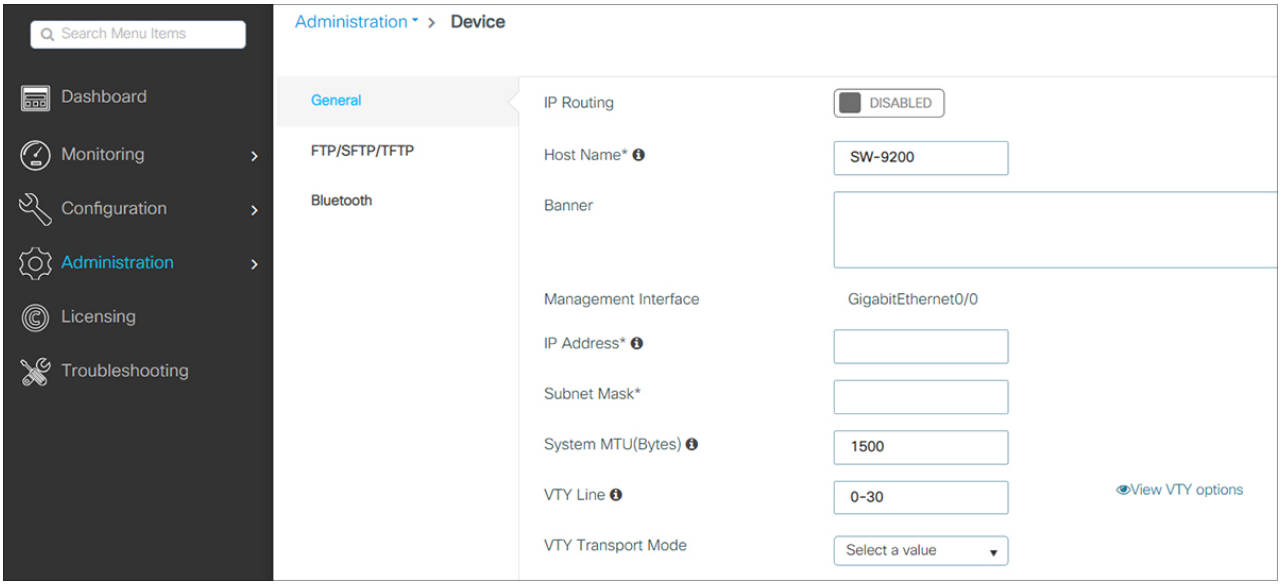
Configuring VTY Lines

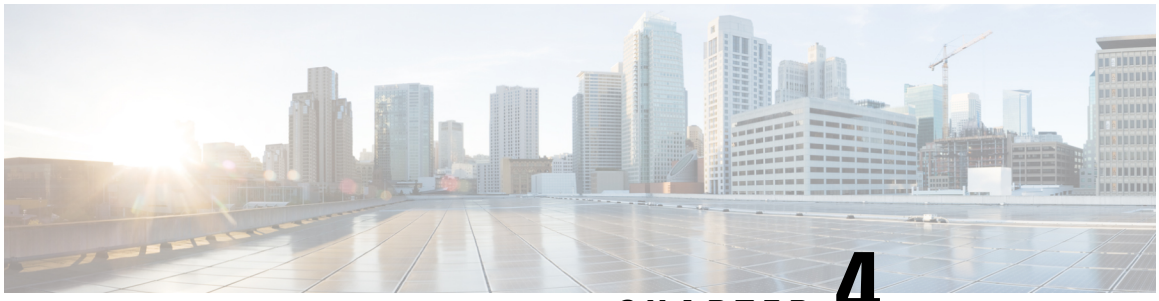
For connecting to the device through Telnet or SSH, the Virtual Terminal Lines or Virtual TeleType (VTY) is used. The number of VTY lines is the maximum number of simultaneous access to the device remotely. If the device is not configured with sufficient number of VTY lines, users might face issues with connecting to the WebUI. You must change the default value for VTY Line, 0-15 (or 0-4 in some models), to 0-30 to allow up to thirty simultaneous sessions.

Step 1 From the WebUI, navigate through **Administration** > **Device** and select the **General** page.

Step 2 In the **VTY Line** field, enter `0-30`.

Figure 17: Configuring VTY Line





CHAPTER 4

Administering the Device

Information About Administering the Device

This section contains the following:

System Time and Date Management

You can manage the system time and date on your device using automatic configuration methods (RTC and NTP), or manual configuration methods.

Note For complete syntax and usage information for the commands used in this section, see the *Cisco IOS Configuration Fundamentals Command Reference* on *Cisco.com*.

System Clock

The basis of the time service is the system clock. This clock runs from the moment the system starts up and keeps track of the date and time.

The system clock can then be set from these sources:

- RTC
- NTP
- Manual configuration

The system clock can provide time to these services:

- User **show** commands
- Logging and debugging messages

The system clock keeps track of time internally based on Coordinated Universal Time (UTC), also known as Greenwich Mean Time (GMT). You can configure information about the local time zone and summer time (daylight saving time) so that the time appears correctly for the local time zone.

The system clock keeps track of whether the time is *authoritative* or not (that is, whether it has been set by a time source considered to be authoritative). If it is not authoritative, the time is available only for display purposes and is not redistributed.

Network Time Protocol

The NTP is designed to time-synchronize a network of devices. NTP runs over User Datagram Protocol (UDP), which runs over IP. NTP is documented in RFC 1305.

An NTP network usually gets its time from an authoritative time source, such as a radio clock or an atomic clock attached to a time server. NTP then distributes this time across the network. NTP is extremely efficient; no more than one packet per minute is necessary to synchronize two devices to within a millisecond of one another.

NTP uses the concept of a *stratum* to describe how many NTP hops away a device is from an authoritative time source. A stratum 1 time server has a radio or atomic clock directly attached, a stratum 2 time server receives its time through NTP from a stratum 1 time server, and so on. A device running NTP automatically chooses as its time source the device with the lowest stratum number with which it communicates through NTP. This strategy effectively builds a self-organizing tree of NTP speakers.

NTP avoids synchronizing to a device whose time might not be accurate by never synchronizing to a device that is not synchronized. NTP also compares the time reported by several devices and does not synchronize to a device whose time is significantly different than the others, even if its stratum is lower.

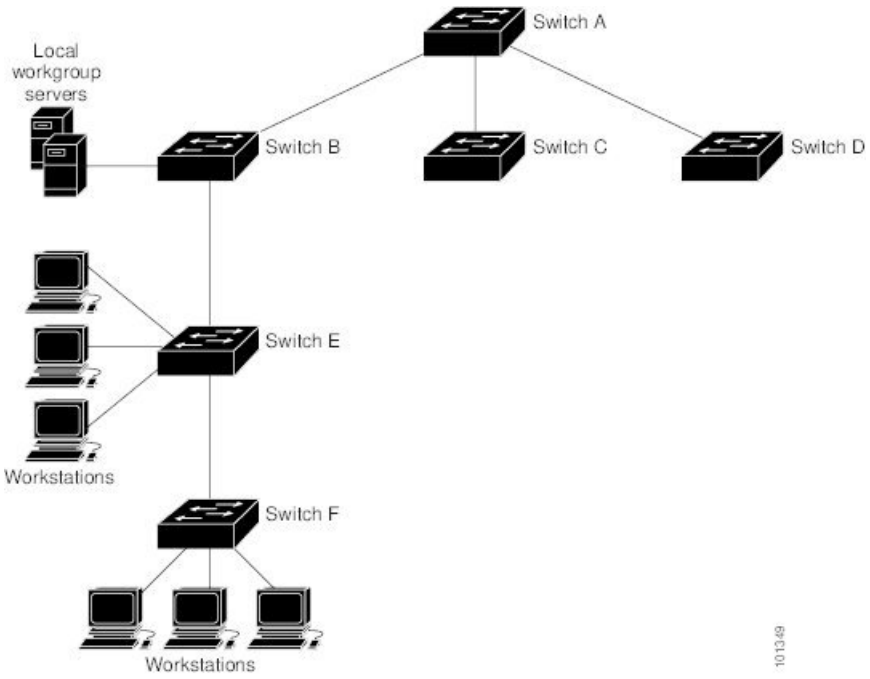
The communications between devices running NTP (known as associations) are usually statically configured; each device is given the IP address of all devices with which it should form associations. Accurate timekeeping is possible by exchanging NTP messages between each pair of devices with an association. However, in a LAN environment, NTP can be configured to use IP broadcast messages instead. This alternative reduces configuration complexity because each device can simply be configured to send or receive broadcast messages. However, in that case, information flow is one-way only.

The time kept on a device is a critical resource; you should use the security features of NTP to avoid the accidental or malicious setting of an incorrect time. Two mechanisms are available: an access list-based restriction scheme and an encrypted authentication mechanism.

Cisco’s implementation of NTP does not support stratum 1 service; it is not possible to connect to a radio or atomic clock. We recommend that the time service for your network be derived from the public NTP servers available on the IP Internet.

The Figure shows a typical network example using NTP. Device A is the primary NTP, with the **Device** B, C, and D configured in NTP server mode, in server association with Device A. Device E is configured as an NTP peer to the upstream and downstream device, Device B and Device F, respectively.

Figure 18: Typical NTP Network Configuration



If the network is isolated from the Internet, Cisco's implementation of NTP allows a device to act as if it is synchronized through NTP, when in fact it has learned the time by using other means. Other devices then synchronize to that device through NTP.

When multiple sources of time are available, NTP is always considered to be more authoritative. NTP time overrides the time set by any other method.

Several manufacturers include NTP software for their host systems, and a publicly available version for systems running UNIX and its various derivatives is also available. This software allows host systems to be time-synchronized as well.

NTP Stratum

NTP uses the concept of a *stratum* to describe how many NTP hops away a device is from an authoritative time source. A stratum 1 time server has a radio or atomic clock directly attached, a stratum 2 time server receives its time through NTP from a stratum 1 time server, and so on. A device running NTP automatically chooses as its time source the device with the lowest stratum number with which it communicates through NTP. This strategy effectively builds a self-organizing tree of NTP speakers.

NTP avoids synchronizing to a device whose time might not be accurate by never synchronizing to a device that is not synchronized. NTP also compares the time reported by several devices and does not synchronize to a device whose time is significantly different than the others, even if its stratum is lower.

NTP Associations

The communications between devices running NTP (known as *associations*) are usually statically configured; each device is given the IP address of all devices with which it should form associations. Accurate timekeeping is possible by exchanging NTP messages between each pair of devices with an association. However, in a LAN environment, NTP can be configured to use IP broadcast messages instead. This alternative reduces

configuration complexity because each device can simply be configured to send or receive broadcast messages. However, in that case, information flow is one-way only.

NTP Security

The time kept on a device is a critical resource; you should use the security features of NTP to avoid the accidental or malicious setting of an incorrect time. Two mechanisms are available: an access list-based restriction scheme and an encrypted authentication mechanism.

Note We do not recommend configuring Message Direct 5 (MD5) authentication. You can use other supported authentication methods for stronger encryption.

NTP Implementation

Implementation of NTP does not support stratum 1 service; it is not possible to connect to a radio or atomic clock. We recommend that the time service for your network be derived from the public NTP servers available on the IP Internet.

If the network is isolated from the Internet, NTP allows a device to act as if it is synchronized through NTP, when in fact it has learned the time by using other means. Other devices then synchronize to that device through NTP.

When multiple sources of time are available, NTP is always considered to be more authoritative. NTP time overrides the time set by any other method.

Several manufacturers include NTP software for their host systems, and a publicly available version for systems running UNIX and its various derivatives is also available. This software allows host systems to be time-synchronized as well.

System Name and Prompt

You configure the system name on the device to identify it. By default, the system name and prompt are *Switch*.

If you have not configured a system prompt, the first 20 characters of the system name are used as the system prompt. A greater-than symbol [>] is appended. The prompt is updated whenever the system name changes.

For complete syntax and usage information for the commands used in this section, see the *Cisco IOS Configuration Fundamentals Command Reference*, *Release 12.4* and the *Cisco IOS IP Command Reference*, *Volume 2 of 3: Routing Protocols*, *Release 12.4*.

Default System Name and Prompt Configuration

The default switch system name and prompt is *Switch*.

DNS

The DNS protocol controls the Domain Name System (DNS), a distributed database with which you can map hostnames to IP addresses. When you configure DNS on your device, you can substitute the hostname for the IP address with all IP commands, such as **ping**, **telnet**, **connect**, and related Telnet support operations.

IP defines a hierarchical naming scheme that allows a device to be identified by its location or domain. Domain names are pieced together with periods (.) as the delimiting characters. For example, Cisco Systems is a commercial organization that IP identifies by a *com* domain name, so its domain name is *cisco.com*. A specific device in this domain, for example, the File Transfer Protocol (FTP) system is identified as *ftp.cisco.com*.

To keep track of domain names, IP has defined the concept of a domain name server, which holds a cache (or database) of names mapped to IP addresses. To map domain names to IP addresses, you must first identify the hostnames, specify the name server that is present on your network, and enable the DNS.

Default DNS Settings

Table 6: Default DNS Settings

Feature	Default Setting
DNS enable state	Enabled.
DNS default domain name	None configured.
DNS servers	No name server addresses are configured.

Login Banners

You can configure a message-of-the-day (MOTD) and a login banner. The MOTD banner is displayed on all connected terminals at login and is useful for sending messages that affect all network users (such as impending system shutdowns).

The login banner is also displayed on all connected terminals. It appears after the MOTD banner and before the login prompts.

Note For complete syntax and usage information for the commands used in this section, see the *Cisco IOS Configuration Fundamentals Command Reference, Release 12.4*.

Default Banner Configuration

The MOTD and login banners are not configured.

MAC Address Table

The MAC address table contains address information that the device uses to forward traffic between ports. All MAC addresses in the address table are associated with one or more ports. The address table includes these types of addresses:

- Dynamic address—A source MAC address that the device learns and then ages when it is not in use.
- Static address—A manually entered unicast address that does not age and that is not lost when the device resets.

The address table lists the destination MAC address, the associated VLAN ID, and port number associated with the address and the type (static or dynamic).

Note For complete syntax and usage information for the commands used in this section, see the command reference for this release.

MAC Address Table Creation

With multiple MAC addresses supported on all ports, you can connect any port on the device to other network devices. The device provides dynamic addressing by learning the source address of packets it receives on each port and adding the address and its associated port number to the address table. As devices are added or removed from the network, the device updates the address table, adding new dynamic addresses and aging out those that are not in use.

The aging interval is globally configured. However, the device maintains an address table for each VLAN, and STP can accelerate the aging interval on a per-VLAN basis.

The device sends packets between any combination of ports, based on the destination address of the received packet. Using the MAC address table, the device forwards the packet only to the port associated with the destination address. If the destination address is on the port that sent the packet, the packet is filtered and not forwarded. The device always uses the store-and-forward method: complete packets are stored and checked for errors before transmission.

MAC Addresses and VLANs

All addresses are associated with a VLAN. An address can exist in more than one VLAN and have different destinations in each. Unicast addresses, for example, could be forwarded to port 1 in VLAN 1 and ports 9, 10, and 1 in VLAN 5.

Each VLAN maintains its own logical address table. A known address in one VLAN is unknown in another until it is learned or statically associated with a port in the other VLAN.

Default MAC Address Table Settings

The following table shows the default settings for the MAC address table.

Table 7: Default Settings for the MAC Address

Feature	Default Setting
Aging time	300 seconds
Dynamic addresses	Automatically learned
Static addresses	None configured

ARP Table Management

To communicate with a device (over Ethernet, for example), the software first must learn the 48-bit MAC address or the local data link address of that device. The process of learning the local data link address from an IP address is called *address resolution*.

The Address Resolution Protocol (ARP) associates a host IP address with the corresponding media or MAC addresses and the VLAN ID. Using an IP address, ARP finds the associated MAC address. When a MAC

address is found, the IP-MAC address association is stored in an ARP cache for rapid retrieval. Then the IP datagram is encapsulated in a link-layer frame and sent over the network. Encapsulation of IP datagrams and ARP requests and replies on IEEE 802 networks other than Ethernet is specified by the Subnetwork Access Protocol (SNAP). By default, standard Ethernet-style ARP encapsulation (represented by the **arpa** keyword) is enabled on the IP interface.

ARP entries added manually to the table do not age and must be manually removed.

For CLI procedures, see the Cisco IOS Release 12.4 documentation on *Cisco.com*.

How to Administer the Device

This section contains the following:

Configuring the Time and Date Manually

System time remains accurate through restarts and reboot, however, you can manually configure the time and date after the system is restarted.

We recommend that you use manual configuration only when necessary. If you have an outside source to which the device can synchronize, you do not need to manually set the system clock.

Setting the System Clock

If you have an outside source on the network that provides time services, such as an NTP server, you do not need to manually set the system clock.

Follow these steps to set the system clock:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Use one of the following: • **clock set** *hh:mm:ss day month year* • **clock set** *hh:mm:ss month day year* **Example:** `Device# clock set 13:32:00 23 March 2013`	Manually set the system clock using one of these formats: • *hh:mm:ss*—Specifies the time in hours (24-hour format), minutes, and seconds. The time specified is relative to the configured time zone. • *day*—Specifies the day by date in the month. • *month*—Specifies the month by name. • *year*—Specifies the year (no abbreviation).

Configuring the Time Zone

Follow these steps to manually configure the time zone:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**clock timezone** *zone hours-offset* [*minutes-offset*] **Example:** `Device(config)# clock timezone AST -3 30`	Sets the time zone. Internal time is kept in Coordinated Universal Time (UTC), so this command is used only for display purposes and when the time is manually set. • *zone*—Enters the name of the time zone to be displayed when standard time is in effect. The default is UTC. • *hours-offset*—Enters the hours offset from UTC. • (Optional) *minutes-offset*—Enters the minutes offset from UTC. This available where the local time zone is a percentage of an hour different from UTC.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Summer Time (Daylight Saving Time)

To configure summer time (daylight saving time) in areas where it starts and ends on a particular day of the week each year, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**clock summer-time** *zone* **date** *date month year hh:mm date month year hh:mm* [*offset*]] **Example:** `Device(config)# clock summer-time PDT date 10 March 2013 2:00 3 November 2013 2:00`	Configures summer time to start and end on specified days every year.
Step 4	**clock summer-time** *zone* **recurring** [*week day month hh:mm week day month hh:mm* [*offset*]] **Example:** `Device(config)# clock summer-time PDT recurring 10 March 2013 2:00 3 November 2013 2:00`	Configures summer time to start and end on the specified days every year. All times are relative to the local time zone. The start time is relative to standard time. The end time is relative to summer time. Summer time is disabled by default. If you specify **clock summer-time** *zone* **recurring** without parameters, the summer time rules default to the United States rules. If the starting month is after the ending month, the system assumes that you are in the southern hemisphere. • *zone*—Specifies the name of the time zone (for example, PDT) to be displayed when summer time is in effect. • (Optional) *week*— Specifies the week of the month (1 to 4, **first**, or **last**). • (Optional) *day*—Specifies the day of the week (Sunday, Monday...). • (Optional) *month*—Specifies the month (January, February...).

	Command or Action	Purpose
		• (Optional) *hh:mm*—Specifies the time (24-hour format) in hours and minutes. • (Optional) *offset*—Specifies the number of minutes to add during summer time. The default is 60.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a System Name

Follow these steps to manually configure a system name:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**hostname** *name* **Example:** `Device(config)# hostname`	Configures a system name. When you set the system name, it is also used as the system prompt. The default setting is Switch.

	Command or Action	Purpose
	remote-users	The name must follow the rules for ARPANET hostnames. They must start with a letter, end with a letter or digit, and have as interior characters only letters, digits, and hyphens. Names can be up to 63 characters.

Step 4	**end**
	Example:
	`remote-users(config)#end`
	`remote-users#`
		Returns to priviliged EXEC mode.

Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Setting Up DNS

If you use the device IP address as its hostname, the IP address is used and no DNS query occurs. If you configure a hostname that contains no periods (.), a period followed by the default domain name is appended to the hostname before the DNS query is made to map the name to an IP address. The default domain name is the value set by the **ip domain name** command in global configuration mode. If there is a period (.) in the hostname, the Cisco IOS software looks up the IP address without appending any default domain name to the hostname.

Follow these steps to set up your switch to use the DNS:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip domain name** *name* **Example:** `Device(config)# ip domain name Cisco.com`	Defines a default domain name that the software uses to complete unqualified hostnames (names without a dotted-decimal domain name). Do not include the initial period that separates an unqualified name from the domain name. At boot time, no domain name is configured; however, if the device configuration comes from a BOOTP or Dynamic Host Configuration Protocol (DHCP) server, then the default domain name might be set by the BOOTP or DHCP server (if the servers were configured with this information).
Step 4	**ip name-server** *server-address1* [*server-address2 ... server-address6*] **Example:** `Device(config)# ip name-server 192.168.1.100 192.168.1.200 192.168.1.300`	Specifies the address of one or more name servers to use for name and address resolution. You can specify up to six name servers. Separate each server address with a space. The first server specified is the primary server. The device sends DNS queries to the primary server first. If that query fails, the backup servers are queried.
Step 5	**ip domain lookup** [**nsap** | **source-interface** *interface*] **Example:** `Device(config)# ip domain-lookup`	(Optional) Enables DNS-based hostname-to-address translation on your device. This feature is enabled by default. If your network devices require connectivity with devices in networks for which you do not control name assignment, you can dynamically assign device names that uniquely identify your devices by using the global Internet naming scheme (DNS).
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Message-of-the-Day Login Banner

You can create a single or multiline message banner that appears on the screen when someone logs in to the device.

Follow these steps to configure a MOTD login banner:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**banner motd** *c message c*
	Example:
	`Device(config)# banner motd #`
	`This is a secure site. Only`
	`authorized users are allowed.`
	`For access, contact technical`
	`support.`
	`#`
		Specifies the message of the day.
		c—Enters the delimiting character of your choice, for example, a pound sign (#), and press the **Return** key. The delimiting character signifies the beginning and end of the banner text. Characters after the ending delimiter are discarded.
		message—Enters a banner message up to 255 characters. You cannot use the delimiting character in the message.

Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Login Banner

You can configure a login banner to be displayed on all connected terminals. This banner appears after the MOTD banner and before the login prompt.

Follow these steps to configure a login banner:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**banner login** *c message c*
	Example:
	`Device(config)# banner login $`
	`Access for authorized users only.`
	`Please enter your username and password.`
	`$`
		Specifies the login message.
		c— Enters the delimiting character of your choice, for example, a pound sign (#), and press the **Return** key. The delimiting character signifies the beginning and end of the banner text. Characters after the ending delimiter are discarded.
		message—Enters a login message up to 255 characters. You cannot use the delimiting character in the message.

Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Managing the MAC Address Table

Changing the Address Aging Time

Follow these steps to configure the dynamic address table aging time:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mac address-table aging-time** [*0* | *10-1000000*] [**routed-mac** | **vlan** *vlan-id*] **Example:** `Device(config)# mac address-table aging-time 500 vlan 2`	Sets the length of time that a dynamic entry remains in the MAC address table after the entry is used or updated. The range is 10 to 1000000 seconds. The default is 300. You can also enter 0, which disables aging. Static address entries are never aged or removed from the table. *vlan-id*—Valid IDs are 1 to 4094.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring MAC Address Change Notification Traps

Follow these steps to configure the switch to send MAC address change notification traps to an NMS host:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server host** *host-addr community-string notification-type* { **informs** | **traps** } {**version** {**1** | **2c** | **3**}} {**vrf** *vrf instance name*} **Example:** `Device(config)# snmp-server host 192.168.10.10 traps private mac-notification`	Specifies the recipient of the trap message. • *host-addr*—Specifies the name or address of the NMS. • **traps** (the default)—Sends SNMP traps to the host. • **informs**—Sends SNMP informs to the host. • **version**—Specifies the SNMP version to support. Version 1, the default, is not available with informs. • *community-string*—Specifies the string to send with the notification operation. Though you can set this string by using the **snmp-server host** command, we recommend that you define this string by using the **snmp-server community** command before using the **snmp-server host** command. • *notification-type*—Uses the **mac-notification** keyword. • **vrf** *vrf instance name*—Specifies the VPN routing/forwarding instance for this host.
Step 4	**snmp-server enable traps mac-notification change** **Example:** `Device(config)# snmp-server enable traps mac-notification change`	Enables the device to send MAC address change notification traps to the NMS.
Step 5	**mac address-table notification change** **Example:** `Device(config)# mac address-table`	Enables the MAC address change notification feature.

	Command or Action	Purpose
	`notification change`	

Step 6	**mac address-table notification change** [**interval** *value*] [**history-size** *value*]
	Example:
	`Device(config)# mac address-table notification change interval 123`
	`Device(config)#mac address-table notification change history-size 100`
		Enters the trap interval time and the history table size.
		• (Optional) **interval** *value*—Specifies the notification trap interval in seconds between each set of traps that are generated to the NMS. The range is 0 to 2147483647 seconds; the default is 1 second.
		• (Optional) **history-size** *value*—Specifies the maximum number of entries in the MAC notification history table. The range is 0 to 500; the default is 1.

Step 7	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/0/2`	Enters interface configuration mode, and specifies the Layer 2 interface on which to enable the SNMP MAC address notification trap.
Step 8	**snmp trap mac-notification change** {**added** | **removed**} **Example:** `Device(config-if)# snmp trap mac-notification change added`	Enables the MAC address change notification trap on the interface. • Enables the trap when a MAC address is **added** on this interface. • Enables the trap when a MAC address is **removed** from this interface.
Step 9	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 10	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring MAC Address Move Notification Traps

When you configure MAC-move notification, an SNMP notification is generated and sent to the network management system whenever a MAC address moves from one port to another within the same VLAN.

Follow these steps to configure the device to send MAC address-move notification traps to an NMS host:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server host** *host-addr* {**traps** | **informs**} {**version** {**1** | **2c** | **3**}} *community-string notification-type* **Example:** `Device(config)# snmp-server host 192.168.10.10 traps private mac-notification`	Specifies the recipient of the trap message. • *host-addr*—Specifies the name or address of the NMS. • **traps** (the default)—Sends SNMP traps to the host. • **informs**—Sends SNMP informs to the host. • **version**—Specifies the SNMP version to support. Version 1, the default, is not available with informs. • *community-string*—Specifies the string to send with the notification operation. Though you can set this string by using the **snmp-server host** command, we recommend that you define this string by using the **snmp-server community** command before using the **snmp-server host** command. • *notification-type*—Uses the **mac-notification** keyword.
Step 4	**snmp-server enable traps mac-notification move** **Example:** `Device(config)# snmp-server enable traps mac-notification move`	Enables the device to send MAC address move notification traps to the NMS.
Step 5	**mac address-table notification mac-move** **Example:** `Device(config)# mac address-table notification mac-move`	Enables the MAC address move notification feature.

	Command or Action	Purpose
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to do next

To disable MAC address-move notification traps, use the **no snmp-server enable traps mac-notification move** global configuration command. To disable the MAC address-move notification feature, use the **no mac address-table notification mac-move** global configuration command.

You can verify your settings by entering the **show mac address-table notification mac-move** privileged EXEC commands.

Configuring MAC Threshold Notification Traps

When you configure MAC threshold notification, an SNMP notification is generated and sent to the network management system when a MAC address table threshold limit is reached or exceeded.

Follow these steps to configure the switch to send MAC address table threshold notification traps to an NMS host:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**snmp-server host** *host-addr* {**traps** | **informs**} {**version** {**1** | **2c** | **3**}} *community-string notification-type* **Example:** `Device(config)# snmp-server host 192.168.10.10 traps private mac-notification`	Specifies the recipient of the trap message. • *host-addr*—Specifies the name or address of the NMS. • **traps** (the default)—Sends SNMP traps to the host. • **informs**—Sends SNMP informs to the host. • **version**—Specifies the SNMP version to support. Version 1, the default, is not available with informs. • *community-string*—Specifies the string to send with the notification operation. You can set this string by using the **snmp-server host** command, but we recommend that you define this string by using the **snmp-server community** command before using the **snmp-server host** command. • *notification-type*—Uses the **mac-notification** keyword.
Step 4	**snmp-server enable traps mac-notification threshold** **Example:** `Device(config)# snmp-server enable traps mac-notification threshold`	Enables MAC threshold notification traps to the NMS.
Step 5	**mac address-table notification threshold** **Example:** `Device(config)# mac address-table notification threshold`	Enables the MAC address threshold notification feature.

Step 6	**mac address-table notification threshold** [**limit** *percentage*] | [**interval** *time*]
	Example:
	`Device(config)# mac address-table notification threshold interval 123`
	`Device(config)# mac address-table notification threshold limit 78`
		Enters the threshold value for the MAC address threshold usage monitoring.
		• (Optional) **limit** *percentage*—Specifies the percentage of the MAC address table use; valid values are from 1 to 100 percent. The default is 50 percent.
		• (Optional) **interval** *time*—Specifies the time between notifications; valid values are greater than or equal to 120 seconds. The default is 120 seconds.

Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 8	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Adding and Removing Static Address Entries

Follow these steps to add a static address:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mac address-table static** *mac-addr* **vlan** *vlan-id* **interface** *interface-id* **Example:** `Device(config)# mac address-table static c2f3.220a.12f4 vlan 4 interface gigabitethernet 1/0/1`	Adds a static address to the MAC address table. • *mac-addr*—Specifies the destination MAC unicast address to add to the address table. Packets with this destination address received in the specified VLAN are forwarded to the specified interface. • *vlan-id*—Specifies the VLAN for which the packet with the specified MAC address is received. Valid VLAN IDs are 1 to 4094. • *interface-id*—Specifies the interface to which the received packet is forwarded. Valid interfaces include physical ports or port channels. For static multicast addresses, you can enter multiple interface IDs. For static unicast addresses, you can enter only one interface at a time, but you can enter the command

	Command or Action	Purpose
		multiple times with the same MAC address and VLAN ID.
Step 4	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Unicast MAC Address Filtering

Follow these steps to configure the device to drop a source or destination unicast static address:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mac address-table static** *mac-addr* **vlan** *vlan-id* **drop** **Example:** `Device(config)# mac address-table static c2f3.220a.12f4 vlan 4 drop`	Enables unicast MAC address filtering and configure the device to drop a packet with the specified source or destination unicast static address. • *mac-addr*—Specifies a source or destination unicast MAC address (48-bit). Packets with this MAC address are dropped. • *vlan-id*—Specifies the VLAN for which the packet with the specified MAC address is received. Valid VLAN IDs are 1 to 4094.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring and Maintaining Administration of the Device

Command	Purpose
clear mac address-table dynamic	Removes all dynamic entries.
clear mac address-table dynamic address *mac-address*	Removes a specific MAC address.
clear mac address-table dynamic interface *interface-id*	Removes all addresses on the specified physical port or port channel.
clear mac address-table dynamic vlan *vlan-id*	Removes all addresses on a specified VLAN.
show clock [*detail*]	Displays the time and date configuration.
show ip igmp snooping groups	Displays the Layer 2 multicast entries for all VLANs or the specified VLAN.
show mac address-table address *mac-address*	Displays MAC address table information for the specified MAC address.
show mac address-table aging-time	Displays the aging time in all VLANs or the specified VLAN.
show mac address-table count	Displays the number of addresses present in all VLANs or the specified VLAN.
show mac address-table dynamic	Displays only dynamic MAC address table entries.
show mac address-table interface *interface-name*	Displays the MAC address table information for the specified interface.
show mac address-table move update	Displays the MAC address table move update information.
show mac address-table multicast	Displays a list of multicast MAC addresses.
show mac address-table notification {change | mac-move | threshold}	Displays the MAC notification parameters and history table.
show mac address-table secure	Displays the secure MAC addresses.

Command	Purpose
show mac address-table static	Displays only static MAC address table entries.
show mac address-table vlan *vlan-id*	Displays the MAC address table information for the specified VLAN.

Configuration Examples for Device Administration

Example: Setting the System Clock

This example shows how to manually set the system clock:

```
Device# clock set 13:32:00 23 July 2013
```

Examples: Configuring Summer Time

This example (for daylight savings time) shows how to specify that summer time starts on March 10 at 02:00 and ends on November 3 at 02:00:

```
Device(config)# clock summer-time PDT recurring PST date
10 March 2013 2:00 3 November 2013 2:00
```

This example shows how to set summer time start and end dates:

```
Device(config)#clock summer-time PST date
20 March 2013 2:00 20 November 2013 2:00
```

Example: Configuring a MOTD Banner

This example shows how to configure a MOTD banner by using the pound sign (#) symbol as the beginning and ending delimiter:

```
Device(config)# banner motd #

This is a secure site. Only authorized users are allowed.
For access, contact technical support.

#

Device(config)#
```

This example shows the banner that appears from the previous configuration:

```
Unix> telnet 192.168.2.15

Trying 192.168.2.15...
```

```
Connected to 192.168.2.15.

Escape character is '^]'.

This is a secure site. Only authorized users are allowed.

For access, contact technical support.

User Access Verification

Password:
```

Example: Configuring a Login Banner

This example shows how to configure a login banner by using the dollar sign ($) symbol as the beginning and ending delimiter:

```
Device(config)# banner login $

Access for authorized users only. Please enter your username and password.

$

Device(config)#
```

Example: Configuring MAC Address Change Notification Traps

This example shows how to specify 192.168.10.10 as the NMS, enable MAC address notification traps to the NMS, enable the MAC address-change notification feature, set the interval time to 123 seconds, set the history-size to 100 entries, and enable traps whenever a MAC address is added on the specified port:

```
Device(config)# snmp-server host 192.168.10.10 traps private mac-notification
Device(config)# snmp-server enable traps mac-notification change
Device(config)# mac address-table notification change
Device(config)# mac address-table notification change interval 123
Device(config)# mac address-table notification change history-size 100
Device(config)# interface gigabitethernet1/2/1
Device(config-if)# snmp trap mac-notification change added
```

Example: Configuring MAC Threshold Notification Traps

This example shows how to specify 192.168.10.10 as the NMS, enable the MAC address threshold notification feature, set the interval time to 123 seconds, and set the limit to 78 per cent:

```
Device(config)# snmp-server host 192.168.10.10 traps private mac-notification
Device(config)# snmp-server enable traps mac-notification threshold
Device(config)# mac address-table notification threshold
Device(config)# mac address-table notification threshold interval 123
Device(config)# mac address-table notification threshold limit 78
```

Example: Adding the Static Address to the MAC Address Table

This example shows how to add the static address c2f3.220a.12f4 to the MAC address table. When a packet is received in VLAN 4 with this MAC address as its destination address, the packet is forwarded to the specified port:

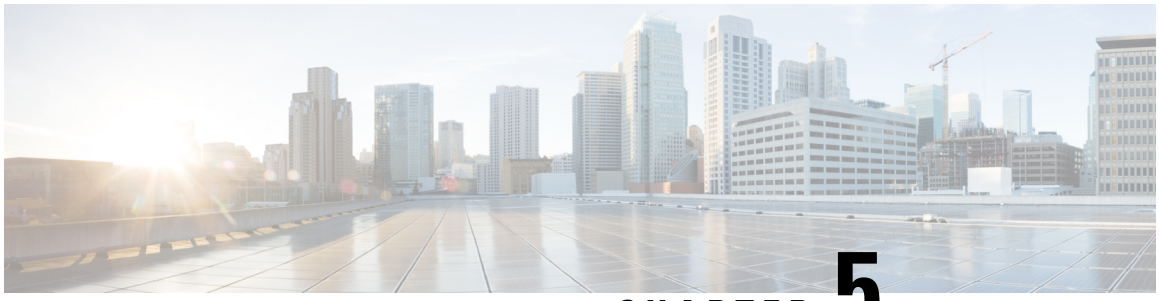
Note You cannot associate the same static MAC address to multiple interfaces. If the command is executed again with a different interface, the static MAC address is overwritten on the new interface.

```
Device(config)# mac address-table static c2f3.220a.12f4 vlan 4 interface gigabitethernet1/1/1
```

Example: Configuring Unicast MAC Address Filtering

This example shows how to enable unicast MAC address filtering and how to configure drop packets that have a source or destination address of c2f3.220a.12f4. When a packet is received in VLAN 4 with this MAC address as its source or destination, the packet is dropped:

```
Device(config)# mac address-table static c2f3.220a.12f4 vlan 4 drop
```



CHAPTER 5

Boot Integrity Visibility

Information About Boot Integrity Visibility

Boot Integrity Visibility allows Cisco's platform identity and software integrity information to be visible and actionable. Platform identity provides the platform's manufacturing installed identity. Software integrity exposes boot integrity measurements that can be used to assess whether the platform has booted trusted code.

During the boot process, the software creates a checksum record of each stage of the bootloader activities.

You can retrieve this record and compare it with a Cisco-certified record to verify if your software image is genuine. If the checksum values do not match, you may be running a software image that is either not certified by Cisco or has been altered by an unauthorized party.

Image Signing and Bootup

The Cisco build servers generate the Cisco IOS XE images. Cisco IOS XE images use the Abraxas image signing system to sign these images securely with the Cisco private RSA keys.

When you copy the Cisco IOS XE image onto a ESS9300, Cisco's ROMMON Boot ROM verifies the image using Cisco release keys. These keys are public keys that correspond to the Cisco release private key that is stored securely on the Abraxas servers. The release key is stored in the ROMMON.

The ESS9300 supports the boot integrity visibility feature. Boot integrity visibility serves as a hardware trust anchor which validates the ROMMON software to ensure that the ROMMON software is not tampered with.

The Cisco IOS XE image is digitally signed during the build time. An SHA-512 hash is generated over the entire binary image file, and then the hash is encrypted with a Cisco RSA 2048-bit private key. The ROMMON verifies the signature using the Cisco public key. If the software is not generated by a Cisco build system, the signature verification fails. The device ROMMON rejects the image and stops booting. If the signature verification is successfully, the device boots the image to the Cisco IOS XE runtime environment.

The ROMMON follows these steps when it verifies a signed Cisco IOS XE image during the bootup:

1. Loads the Cisco IOS XE image into the CPU memory.

2. Examines the Cisco IOS XE package header.
3. Runs a non-secure integrity check on the image to ensure that there is no unintentional file corruption from the disk or TFTP. This is performed using a non-secure SHA-1 hash.
4. Copies the Cisco's RSA 2048-bit public release key from the ROMMON storage and validates that the Cisco's RSA 2048-bit public release key is not tampered.
5. Extracts the Code Signing signature (SHA-512 hash) from the package header and verifies it using Cisco's RSA 2048-bit public release key.
6. Performs the Code Signing validation by calculating the SHA-512 hash of the Cisco IOS XE package and compares it with the Code Signing signature. The Signed package is now validated.
7. Examines the Cisco IOS XE package header to validate the platform type and CPU architecture for compatibility.
8. Extracts the Cisco IOS XE software from the Cisco IOS XE package and boots it.

Note In above process, step 3 is a non-secure check of the image which is intended to confirm the image against inadvertent corruption due to disk errors, file transfer errors, or copying errors. This is not part of the image code signing. This check is not intended to detect deliberate image tampering.

Image Code Signing validation occurs in step 4, 5, and 6. This is a secure code signing check of the image using an SHA-512 hash that is encrypted with a 2048-bit RSA key. This check is intended to detect deliberate image tampering.

Verifying the Software Image and Hardware

This task describes how to retrieve the checksum record that was created during a switch bootup. Enter the following commands in privileged EXEC mode.

Note On executing the following commands, you might see the message **% Please Try After Few Seconds** displayed on the CLI. This does not indicate a CLI failure, but indicates setting up of underlying infrastructure required to get the required output. We recommend waiting for a few minutes and then try the command again.

The messages **% Error retrieving SUDI certificate** and **% Error retrieving integrity data** signify a real CLI failure.

Procedure

	Command or Action	Purpose
Step 1	**show platform sudi certificate** [**sign** [**nonce** *nonce*]] **Example:** `Device# show platform sudi certificate sign nonce`	Displays checksum record for the specific SUDI. • (Optional) **sign** - Show signature • (Optional) **nonce** - Enter a nonce value

	Command or Action	Purpose
	`123`	
Step 2	**show platform integrity** [**sign** [**nonce** *nonce*]] **Example:** `Device# show platform integrity sign nonce 123`	Displays checksum record for boot stages. • (Optional) **sign** - Show signature • (Optional) **nonce** - Enter a nonce value

Verifying Platform Identity and Software Integrity

Verifying Platform Identity

The following example displays the Secure Unique Device Identity (SUDI) chain in PEM format. Encoded into the SUDI is the Product ID and Serial Number of each individual device such that the device can be uniquely identified on a network of thousands of devices. The first certificate is the Cisco Root CA 2048 and the second is the Cisco subordinate CA (ACT2 SUDI CA). Both certificates can be verified to match those published on https://www.cisco.com/security/pki/. The third is the SUDI certificate.

```
Device# show platform sudi certificate sign nonce 123
-----BEGIN CERTIFICATE-----
MIIDQzCCAiugAwIBAgIQX/h7KCtU3I1CoxW1aMmt/zANBgkqhkiG9w0BAQUFADA1
MRYwFAYDVQQKEw1DaXNjbyBTeXN0ZW1zMRswGQYDVQQDExJDaXNjbyBSb290IENB
IDIwNDgwHhcNMDQwNTE0MjAxNzEyWhcNMjkwNTE0MjAyNTQyWjA1MRYwFAYDVQQK
Ew1DaXNjbyBTeXN0ZW1zMRswGQYDVQQDExJDaXNjbyBSb290IENBIDIwNDgwggEg
MA0GCSqGSIb3DQEBAQUAA4IBDQAwggEIAoIBAQCwmrmrp68Kd6ficba0ZmKUeIhH
xmJVhEAyv8CrLqUccda8bnuoqrpu0hWISEWdovyD0My5jOAmaHBKeN8hF570YQXJ
FcjPFto1YYmUQ6iEqDGYeJu5Tm8sUxJszR2tKyS7McQr/4NEb7Y9JHcJ6r8qqB9q
VvYgDxFUl4F1pyXOWWqCZe+36ufijXWLbvLdT6ZeYpzPEApk0E5tzivMW/VgpSdH
jWn0f84bcN5wGyDWbs2mAag8EtKpP6BrXruOIIt6keO1aO6g58QBdKhTCytKmg9l
Eg6CTY5j/e/rmxrbU6YTYK/CfdfHbBcl1HP7R2RQgYCUTOG/rksc35LtLgXfAgED
o1EwTzALBgNVHQ8EBAMCAYYwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQUJ/PI
FR5umgIJFq0roIlgX9p7L6owEAYJKwYBBAGCNxUBBAMCAQAwDQYJKoZIhvcNAQEF
BQADggEBAJ2dhISjQal8dwy3U8pORFBi71R803UXHOjgxkhLtv5MOhmBVrBW7hmW
Yqpao2TB9k5UM8Z3/sUcuuVdJcr18JOagxEu5sv4dEX+5wW4q+ffy0vhN4TauYuX
cB7w4ovXsNgOnbFp1iqRe6lJT37mjpXYgyc81WhJDtSd9i7rp77rMKSsH0T8lasz
Bvt9YAretIpjsJyp8qS5UwGH0GikJ3+r/+n6yUA4iGe0OcaEb1fJU9u6ju7AQ7L4
CYNu/2bPPu8Xs1gYJQk0XuPL1hS27PKSb3TkL4Eq1ZKR4OCXPDJoBYVL0fdX4lId
kxpUnwVwwEpxYB5DC2Ae/qPOgRnhCzU=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIEPDCCAySgAwIBAgIKYQlufQAAAAAADDANBgkqhkiG9w0BAQUFADA1MRYwFAYD
VQQKEw1DaXNjbyBTeXN0ZW1zMRswGQYDVQQDExJDaXNjbyBSb290IENBIDIwNDgw
HhcNMTEwNjMwMTc1NjU3WhcNMjkwNTE0MjAyNTQyWjAnMQ4wDAYDVQQKEwVDaXNj
bzEVMBMGA1UEAxMMQUNUMiBTVURJIENBMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8A
MIIBCgKCAQEA0m5l3THIxA9tN/hS5qR/6UZRpdd+9aE2JbFkNjht6gfHKd477AkS
5XAtUs5oxDYVt/zEbslZq3+LR6qrqKKQVu6JYvH05UYLBqCj38s76NLk53905Wzp
9pRcmRCPuX+a6tHF/qRuOiJ44mdeDYZo3qPCpxzprWJDPclM4iYKHumMQmqmgmg+
xghHIooWS80BOcdiynEbeP5rZ7qRuewKMpl1TiI3WdBNjZjnpfjg66F+P4SaDkGb
BXdGj13oVeF+EyFWLrFjj97fL2+8oauV43Qrvnf3d/GfqXj7ew+z/sXlXtEOjSXJ
URsyMEj53Rdd9tJwHky8neapszS+r+kdVQIDAQABo4IBWjCCAVYwCwYDVR0PBAQD
AgHGMB0GA1UdDgQWBBRI2PHxwnDVW7t8cwmTr7i4MAP4fzAfBgNVHSMEGDAWgBQn
88gVHm6aAgkWrSugiWBf2nsvqjBDBgNVHR8EPDA6MDigNqA0hjJodHRwOi8vd3d3
LmNpc2NvLmNvbS9zZWN1cml0eS9wa2kvY3JsL2NyY2EyMDQ4LmNybDBQBggrBgEF
BQcBAQREMEIwQAYIKwYBBQUHMAKGNGh0dHA6Ly93d3cuY2lzY28uY29tL3NlY3Vy
aXR5L3BraS9jZXJ0cy9jcmNhMjA0OC5jZXIwXAYDVR0gBFUwUzBRBgorBgEEAQkV
```

```
AQwAMEMwQQYIKwYBBQUHAgEWNWh0dHA6Ly93d3cuY2lzY28uY29tL3NlY3VyaXR5
L3BraS9wb2xpY2llcy9pbmRleC5odG1sMBIGA1UdEwEB/wQIMAYBAf8CAQAwDQYJ
KoZIhvcNAQEFBQADggEBAGh1qclr9tx4hzWgDERm371yeuEmqcIfi9b9+GbMSJbi
ZHc/CcCl0lJu0a9zTXA9w47H9/t6leduGxb4WeLxcwCiUgvFtCa51Iklt8nNbcKY
/4dw1ex+7amATUQO4QggIE67wVIPu6bgAE3Ja/nRS3xKYSnj8H5TehimBSv6TECi
i5jUhOWryAK4dVo8hCjkjEkzu3ufBTJapnv89g9OE+H3VKM4L+/KdkUO+52djFKn
hyl47d7cZR4DY4LIuFM2P1As8YyjzoNpK/urSRI14WdIlplR1nH7KNDl5618yfVP
0IFJZBGrooCRBjOSwFv8cpWCbmWdPaCQT2nwIjTfY8c=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIDgTCCAmmgAwIBAgIEAp4UYzANBgkqhkiG9w0BAQsFADAnMQ4wDAYDVQQKEwVD
aXNjbzEVMBMGA1UEAxMMQUNUMiBTVURJIENBMB4XDTE4MDYwNTAzNDUwNVoXDTI5
MDUxNDIwMjU0MVowbTEpMCcGA1UEBRMgUElEOkM5MjAwTC0yNFQtNEcgU046SlBH
MjIwMjAwQTgxDjAMBgNVBAoTBUNpc2NvMRgwFgYDVQQLEw9BQ1QtMiBMaXRlIFNV
REkxFjAUBgNVBAMTDUM5MjAwTC0yNFQtNEcwggEiMA0GCSqGSIb3DQEBAQUAA4IB
DwAwggEKAoIBAQDBm2Dg0GWQ18wLTKxeCt87DL8KlRbx8DblIigHjzebBXMpx7Ja
6Cp+kwRrIWGi5AmNmV7jZ2ZLj+vFVzBQ9eGM+6LdNg18c6nqmSmnuXMerDlUEMMK
bkFl4ydn1EIMoWpCArbgz+/zaLM2A5bpQXVndiKq1v0NA2Pgvqdxbm+8AELdDG/D
3SQ1anOja+yH5vu3NjyMJfqtjzk+n/ILp9iZMWzcA+O6E8KC5FclR2cfvWlQvoFM
ZEWmHdhHPtsnN+4hhmDeurgeM0S+xIvzZq0H7PxS0kT4vYQ9xWQEwavJAL44k0uY
JxKP6bDNssSLZ2s4/2OBsODjyBhb0GwrOAHdAgMBAAGjbzBtMA4GA1UdDwEB/wQE
AwIF4DAMBgNVHRMBAf8EAjAAME0GA1UdEQRGMESgQgYJKwYBBAEJFQIDoDUTM0No
aXBJRD1RRGx6T0FZUHQwRTJJRVFFQUFjQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFB
PTANBgkqhkiG9w0BAQsFAAOCAQEAgLUxZfNmrXZ6ZMGX69dDPkmvp9cFqXR538LF
PdypCRuSk20GF8OeDUOsuIi4mbB87JSOWvLomdBtXdnxzRu4kPZNFz/7pjAVRT3R
gwMMyiEnDWQSvy7e4SZmyVgej55e3hTW/LTeU8lCE0KRoYGDce5Phv2zdHtIsXrV
XsY+Fropfntt1FV9qqDskDWcKf0bos6VsyWUpSCEGqF7LfNnBTKYvXUUmkXHKf/d
W5HgrYt6bQ/h/+0EP+MY2wpAiWMCfX6F+xW20vZfK8NzNesieB38IvuTkgefhz2s
yGCOavAxqGd0j7atcRpdrJt9+KM9Vwuy4VJZgK/t1fmTL4cawQ==
-----END CERTIFICATE-----

Signature version: 1
Signature:
2AF6EDA39A17403F621BB94E824C4FE00C19D31BF9DFAC00747C0187DF404077505
6E0AE63520E763A5DF0FAEB4FA2B5BF2F9CCF3E8EDE25E7510573CF6669029FC4B22
E4A15841EDA48075ADCBEED6E003C2B6637E0D4ADDBA3754AA1F2EE6AC36AE6FCE00
DD075908148A25767C86F8121AF0DE95534046418A6771323C02801CEB6F412C131AA
31EAB538B39B7143114AB033A3BAD1EA5F02D9A4AF89806BED6EDA0847B310FABD224
7626A9FF150A8D3A82323E17C3DADECF3E2701B03336EA32C371CE88689892423F725
D14919BF777DA60A823008E39A19FF65B8226D8CF4D415212C72A2814A7A7E50CCC759
483B97C1704977B62191741EA5096BE9
```

The optional RSA 2048 signature is across the three certificates, the signature version and the user-provided nonce.

```
RSA PKCS#1v1.5 Sign {<Nonce (UINT64)> || <Signature Version (UINT32)> || <Cisco Root CA
2048 cert (DER)> ||
<Cisco subordinate CA (DER)> || <SUDI certificate (DER)> }
```

Cisco management solutions are equipped with the ability to interpret the above output. However, a simple script using OpenSSL commands can also be used to display the identity of the platform and to verify the signature, thereby ensuring its Cisco unique device identity.

```
[linux-host:~]openssl x509 -in sudicert.pem -subject -noout
subject= /serialNumber=PID:ESS9300 SN:FDO1946BG05/O=Cisco/OU=ACT-2 Lite SUDI/CN=ESS9300
```

Verifying Software Integrity

The following example displays the checksum record for the boot stages. The hash measurements are displayed for each of the three stages of software successively booted. These hashes can be compared against Cisco-provided reference values. An option to sign the output gives a verifier the ability to ensure the output is genuine and is not altered. A nonce can be provided to protect against replay attacks.

Note Boot integrity hashes are not MD5 hashes. For example, if you run **verify /md5 ess9300_iosxe.17.04.01.SPA.bin** command for the bundle file, the hash will not match.

The following is a sample output of the **show platform integrity sign nonce 123** command in install mode. This output includes measurements of each installed package file.

```
Device#show platform integrity sign nonce 123
Platform: ess9300
Boot 0 Version: SBOOT0.v27
Boot 0 Hash:
EE98DCD0D6AEA85C8891039F649664FCC3CF709CCFC7A6F248C9D5BA8463528F
Boot Loader Version: System Bootstrap, Version 10.2, DEVELOPMENT SOFTWARE
Boot Loader Hash:
922B87E7A153A79EE9AE3711A1FDE2313C9996F21032F8A1E7EF4935D3E742765 7E4CDEE537E7B3C50E84121C00ED2556786A4EE155D8C0AFF67F63F1A69B
OS Version: 17.04.01
OS Hashes:
ess9300_lite-rpbase.17.04.01.SPA.pkg :
DDD155C1DEFDB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0
ess9300_lite-rpboot.17.04.01.SPA.pkg :
AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFDB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFB03EB0C64057
ess9300_lite-srdriver.17.04.01.SPA.pkg :
4EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFDB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFB03EB0C64057AD6A9673E211
ess9300_lite-webui.17.04.01.SPA.pkg :
CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFDB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFB03EB0C64057AD6A9673E2114EA7
ess9300-wlc.17.04.01.SPA.pkg :
AA7ED0AE935CB0ED84E0DDD155C1DEFDB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCA
PCR0: 750E5D2EDAE6E3A68050638E0BFD8619BE4EA13066025D39DF79408719F5177E
PCR8: EB6E739A63F53E703B6CDAF3F6188833CEF6D32E2F726006B9AA34E1E73048C4
Signature version: 1
Signature:
[redacted]
```

The following is a sample output of the **show platform integrity sign nonce 123** command in bundle mode. This output includes measurements of the bundle file and each installed package.

```
Device# show platform integrity sign nonce 123
Platform: ESS9300
Boot 0 Version: SBOOT0.v27
Boot 0 Hash:
EE98DCD0D6AEA85C8891039F649664FCC3CF709CCFC7A6F248C9D5BA8463528F
Boot Loader Version: System Bootstrap, Version 10.2, DEVELOPMENT SOFTWARE
Boot Loader Hash:
922B87E7A153A79EE9AE3711A1FDE2313C9996F21032F8A1E7EF4935D3E742765 7E4CDEE537E7B3C50E84121C00ED2556786A4EE155D8C0AFF67F63F1A69B
OS Version: 17.04.01
OS Hashes:
ess9300_lite_iosxe.17.04.01.SPA.bin :
F4CAD08EE1EF841C3A2E3ED854082F08F3CEA9336F38E4569D4D8E15AD15E36E3922AC8B4D0CDB63E2806D6A1EDAE7839D9D08CD7E366A49ED648C113440
ess9300_lite-rpbase.17.04.01.SPA.pkg :
DDD155C1DEFDB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0
ess9300_lite-rpboot.17.04.01.SPA.pkg :
```

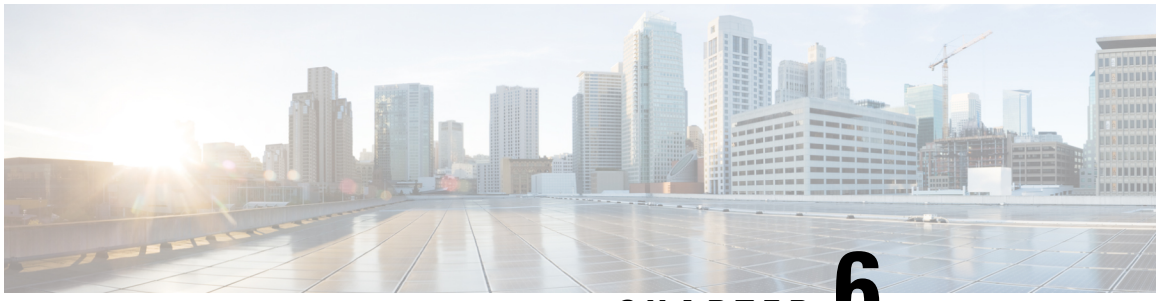
```
AD6A9673E2114EA7CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057
ess9300_lite-srdriver.17.04.01.SPA.pkg :
4EA7CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E211
ess9300_lite-webui.17.04.01.SPA.pkg :
CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E2114EA7
ess9300-wlc.17.04.01.SPA.pkg :
AA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCA
PCR0: 750E5D2EDAE6E3A68050638E0BFD8619BE4EA13066025D39DF79408719F5177E
PCR8: EB6E739A63F53E703B6CDAF3F6188833CEF6D32E2F726006B9AA34E1E73048C4
Signature version: 1
Signature:
[redacted]
```

Verifying Image Signing

The following example displays the secure code signing check of the image during bootup using an SHA-512 hash.

```
switch:boot flash:packages.conf
boot: attempting to boot from [flash:packages.conf]
boot: reading file packages.conf
#
Performing Integrity Check ...
boot: parsed image from conf file: ess9300-rpboot.17.04.01.SSA.pkg
[redacted]


Loading image in Verbose mode: 1


Image Base is: 0x100099000
Image Size is: 0x2C83487
Package header rev 3 structure detected
Package type:30001, flags:0x0
IsoSize = 0
Parsing package TLV info:
000: 000000090000001D4B45595F544C565F -         KEY_TLV_
010: 5041434B4147455F434F4D5041544942 - PACKAGE_COMPATIB
020: 494C4954590000000000000090000000B - ILITY
030: 4652555F52505F545950450000000009 - FRU_RP_TYPE
040: 000000184B45595F544C565F5041434B -     KEY_TLV_PACK
050: 4147455F424F4F544152434800000009 - AGE_BOOTARCH
060: 0000000E415243485F693638365F5459 -     ARCH_i686_TY
070: 5045000000000009000000144B45595F - PE          KEY_
080: 544C565F424F4152445F434F4D504154 - TLV_BOARD_COMPAT
090: 0000000900000010424F4152445F6361 -         BOARD_ca
0A0: 74396B5F5459504500000000900000018 - t9k_TYPE
0B0: 4B45595F544C565F43525950544F5F4B - KEY_TLV_CRYPTO_K
0C0: 4559535452494E470000000900000004 - EYSTRING

TLV: T=9, L=29, V=KEY_TLV_PACKAGE_COMPATIBILITY
TLV: T=9, L=11, V=FRU_RP_TYPE
TLV: T=9, L=24, V=KEY_TLV_PACKAGE_BOOTARCH
TLV: T=9, L=14, V=ARCH_i686_TYPE
TLV: T=9, L=20, V=KEY_TLV_BOARD_COMPAT
```

```
TLV: T=9, L=16, V=BOARD_ess9300_TYPE
TLV: T=9, L=24, V=KEY_TLV_CRYPTO_KEYSTRING
TLV: T=9, L=4, V=none
TLV: T=9, L=11, V=CW_BEGIN=$$
TLV: T=9, L=17, V=CW_FAMILY=$ess9300$
TLV: T=9, L=74, V=CW_IMAGE=$ess9300-rpboot.17.02.01.SSA.pkg$
TLV: T=9, L=20, V=CW_VERSION=$17.2.01$
IOS version is 17.2.1
TLV: T=9, L=53, V=CW_FULL_VERSION=$17.2.01.0.869.1580816579..Amsterdam$
TLV: T=9, L=52, V=CW_DESCRIPTION=$Cisco IOS Software, IOS-XE Software$
TLV: T=9, L=9, V=CW_END=$$
Found DIGISIGN TLV type 12 length = 392
RSA Self Test Passed

Expected hash:
DDAF35A193617ABACC417349AE204131
12E6FA4E89A97EA20A9EEEE64B55D39A
2192992A274FC1A836BA3C23A3FEEBBD
454D4423643CE80E2A9AC94FA54CA49F

Obtained hash:
DDAF35A193617ABACC417349AE204131
12E6FA4E89A97EA20A9EEEE64B55D39A
2192992A274FC1A836BA3C23A3FEEBBD
454D4423643CE80E2A9AC94FA54CA49F
Sha512 Self Test Passed
Found package arch type ARCH_i686_TYPE
Found package FRU type FRU_RP_TYPE
Performing Integrity Check ...

RSA Signed DEVELOPMENT Image Signature Verification Successful.
```



CHAPTER 6

Performing Device Setup Configuration

Restrictions for Performing Device Setup Configuration

- Subpackage software installation is not supported.

Information About Performing Device Setup Configuration

The following sections provide information about how to perform a device setup configuration, including IP address assignments and Dynamic Host Configuration Protocol (DHCP) auto configuration.

Device Boot Process

The normal boot process involves the operation of the boot loader software and includes these activities:

- Performs low-level CPU initialization. This process initializes the CPU registers that control where physical memory is mapped, the quantity and speed of the physical memory, and so forth.
- Initializes the file systems on the system board.
- Loads a default operating system software image into memory and boots up the device.
- Performs power-on self-test (POST) for the CPU subsystem and tests the system DRAM. As part of POST, the following tests are performed:
  - MAC loopback test to verify the data path between the CPU and network ports.
  - Thermal test to verify the temperature reading from the device sensor.

The boot loader provides access to the file systems before the operating system is loaded. Normally, the boot loader is used only to load, decompress, and start the operating system. After the boot loader gives the operating system control of the CPU, the boot loader is not active until the next system reset or power-on.

Before you can assign device information, make sure you have connected a PC or terminal to the console port or a PC to the Ethernet management port, and make sure you have configured the PC or terminal-emulation software baud rate and character format to match these of the device console port:

- Baud rate default is 9600.
- Data bits default is 8.

Note If the data bits option is set to 8, set the parity option to none.

- Stop bits default is 2 (minor).
- Parity settings default is none.

Software Install Overview

The Software Install feature provides a uniform experience across different types of upgrades, such as full image install and Software Maintenance Upgrade (SMU)

The Software Install feature facilitates moving from one version of the software to another version in install mode. Use the **install** command in privileged EXEC mode to install or upgrade a software image. You can also downgrade to a previous version of the software image, using the install mode.

The method that you use to upgrade Cisco IOS XE software depends on whether the switch is running in install mode or in bundle mode. In bundle mode or consolidated boot mode, a .bin image file is used from a local or remote location to boot the device. In the install boot mode, the bootloader uses the packages.conf file to boot up the device.

The following software install features are supported on your switch:

- Software bundle installation.
- Software rollback to a previously installed package set.
- Emergency installation in the event that no valid installed packages reside on the boot flash.

Software Boot Modes

Your device supports two modes to boot the software packages:

- Installed mode
- Bundle mode

Installed Boot Mode

You can boot your device in installed mode by booting the software package provisioning file that resides in flash:

```
Switch: boot flash:packages.conf
```

Note The packages.conf file for particular release is created on following the install workflow described in the section, *Installing a Software Package*.

The provisioning file contains a list of software packages to boot, mount, and run. The ISO file system in each installed package is mounted to the root file system directly from flash.

Note The packages and provisioning file used to boot in installed mode must reside in flash. Booting in installed mode from usbflash0: or tftp: is not supported.

Installing a Software Package

You can install, activate, and commit a software package using a single command or using separate commands. This task shows how to use the **install add file activate commit** command for installing a software package.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**install add file tftp:** *filename* [**activate commit**] **Example:** `Device# install add file flash:ess9300_lite_iosxe.17.04.01.SPA.bin activate commit`	Copies the software install package from a remote location (via FTP, HTTP, HTTPs, TFTP) to the device, performs a compatibility check for the platform and image versions, activates the software package, and makes the package persistent across reloads. • This command extracts the individual components of the .bin file into sub-packages and packages.conf file. • The device reloads after executing this command.
Step 3	**exit** **Example:** `Device# exit`	Exits privileged EXEC mode and returns to user EXEC mode.

Managing the Update Package

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**install add file tftp:** *filename* **Example:** `Device# install add file tftp://172.16.0.1//tftpboot/folder1/ess9300_iosxe.17.04.01.SPA.bin`	Copies the software install package from a remote location (via FTP, HTTP, HTTPs, TFTP) to the device, and performs a compatibility check for the platform and image versions. • This command extracts the individual components of the .bin file into sub-packages and packages.conf file.
Step 3	**install activate** [**auto-abort-timer**] **Example:** `Device# install activate`	Activates the added software install package, and reloads the device. • When doing a full software install, do not provide a package filename. • The **auto-abort-timer** keyword, automatically rolls back the software image activation. The automatic timer is triggered after the new image is activated. If the timer expires prior to the issuing of the **install commit** command, then the install process is automatically terminated. The device reloads, and boots up with a previous version of the software image.

	Command or Action	Purpose
Step 4	**install abort** **Example:** `Device# install abort`	(Optional) Terminates the software install activation, and rolls back to the version that was running before current installation procedure. • You can use this command only when the image is in an activated state; and not when the image is in a committed state.
Step 5	**install commit** **Example:** `Device# install commit`	Makes the changes persistent over reload. • The **install commit** command completes the new image installation. Changes are persistent across reloads until the auto-abort timer expires.
Step 6	**install rollback to committed** **Example:** `Device# install rollback to committed`	(Optional) Rolls back the update to the last committed version.
Step 7	**install remove** {**file** *filesystem: filename* | **inactive**} **Example:** `Device# install remove inactive`	(Optional) Deletes all unused and inactive software installation files.
Step 8	**show install summary** **Example:** `Device# show install summary`	Displays information about the active package. • The output of this command varies according to the **install** commands that are configured.

Bundle Boot Mode

You can boot your device in bundle boot mode by booting the bundle (.bin) file:

```
switch:  boot flash:cat9k_lite_iosxe.16.09.02.SPA.bin
```

The provisioning file contained in a bundle is used to decide which packages to boot, mount, and run. Packages are extracted from the bundle and copied to RAM. The ISO file system in each package is mounted to the root file system.

Unlike install boot mode, additional memory that is equivalent to the size of the bundle is used when booting in bundle mode.

Unlike install boot mode, bundle boot mode is available from several locations:

- flash:
- usbflash0:
- tftp:

Booting a Device in Bundle Mode

There are several methods by which you can boot the device — either by copying the bin file from the TFTP server and then boot the device, or by booting the device straight from flash or USB flash using the commands **boot flash:<image.bin>** or **boot usbflash0:<image.bin>** .

The following procedure explains how to boot the device from the TFTP server in the bundle mode.

Procedure

	Command or Action	Purpose
Step 1	**switch:BOOT=**<*source path of .bin file*> **Example:** `switch:`	Sets the boot parameters.
Step 2	**boot** **Example:** `switch:boot`	Boots the device.
Step 3	**show version**	(Optional) Displays the version of the image installed.

Changing the Boot Mode

To change a device running in bundle boot mode to install mode, set the boot variable to flash:packages.conf, and execute the **install add file flash:ess9300.bin activate commit** command. After the command is executed, the device reboots in install boot mode.

Installing the Software Package

You can install the software package on a device by using the **install add** commands in privileged EXEC mode.

The **install add** command copies the software package from a local or remote location to the device. The location can be FTP, HTTP, HTTPs, or TFTP. The command extracts individual components of the .bin file into sub-packages and packages.conf file. It also validates the file to ensure that the image file is specific to the platform.

Terminating a Software Install

You can terminate the activation of a software image in the following ways:

- Using the **install activate auto-abort-timer** command. When the device reloads after activating a new image, the auto-abort-timer is triggered. If the timer expires before issuing the **install commit** command, then the installation process is terminated; the device reloads again and boots up with the previous version of the software image.

  Use the **install auto-abort-timer stop** command to stop this timer.

- Using the **install abort** command. This command rolls back to the version that was running before installing the new software. Use this command before issuing the **install commit** command.

Devices Information Assignment

You can assign IP information through the device setup program, through a DHCP server, or manually.

Use the device setup program if you want to be prompted for specific IP information. With this program, you can also configure a hostname and an enable secret password.

It gives you the option of assigning a Telnet password (to provide security during remote management) and configuring your switch as a command or member switch of a cluster or as a standalone switch.

Use a DHCP server for centralized control and automatic assignment of IP information after the server is configured.

Note If you are using DHCP, do not respond to any of the questions in the setup program until the device receives the dynamically assigned IP address and reads the configuration file.

If you are an experienced user familiar with the device configuration steps, manually configure the device. Otherwise, use the setup program described in section Device Boot Process, on page 68.

Default Switch Information

Table 8: Default Switch Information

Feature	Default Setting
IP address and subnet mask	No IP address or subnet mask are defined.
Default gateway	No default gateway is defined.
Enable secret password	No password is defined.
Hostname	The factory-assigned default hostname is device.
Telnet password	No password is defined.
Cluster command switch functionality	Disabled.
Cluster name	No cluster name is defined.

DHCP-Based Autoconfiguration Overview

DHCP provides configuration information to Internet hosts and internetworking devices. This protocol consists of two components: one for delivering configuration parameters from a DHCP server to a device and an operation for allocating network addresses to devices. DHCP is built on a client-server model, in which

designated DHCP servers allocate network addresses and deliver configuration parameters to dynamically configured devices. The device can act as both a DHCP client and a DHCP server.

During DHCP-based autoconfiguration, your device (DHCP client) is automatically configured at startup with IP address information and a configuration file.

With DHCP-based autoconfiguration, no DHCP client-side configuration is needed on your device. However, you need to configure the DHCP server for various lease options associated with IP addresses.

If you want to use DHCP to relay the configuration file location on the network, you might also need to configure a Trivial File Transfer Protocol (TFTP) server and a Domain Name System (DNS) server.

The DHCP server for your device can be on the same LAN or on a different LAN than the device. If the DHCP server is running on a different LAN, you should configure a DHCP relay device between your device and the DHCP server. A relay device forwards broadcast traffic between two directly connected LANs. A router does not forward broadcast packets, but it forwards packets based on the destination IP address in the received packet.

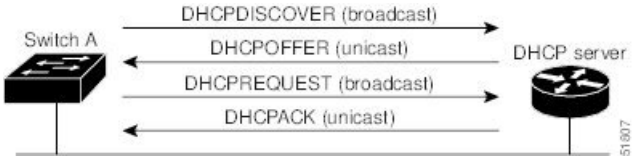
DHCP-based autoconfiguration replaces the BOOTP client functionality on your device.

DHCP Client Request Process

When you boot up your device, the DHCP client is invoked and requests configuration information from a DHCP server when the configuration file is not present on the device. If the configuration file is present and the configuration includes the **ip address dhcp** interface configuration command on specific routed interfaces, the DHCP client is invoked and requests the IP address information for those interfaces.

This is the sequence of messages that are exchanged between the DHCP client and the DHCP server.

Figure 19: DHCP Client and Server Message Exchange



The client, Device A, broadcasts a DHCPDISCOVER message to locate a DHCP server. The DHCP server offers configuration parameters (such as an IP address, subnet mask, gateway IP address, DNS IP address, a lease for the IP address, and so forth) to the client in a DHCPOFFER unicast message.

In a DHCPREQUEST broadcast message, the client returns a formal request for the offered configuration information to the DHCP server. The formal request is broadcast so that all other DHCP servers that received the DHCPDISCOVER broadcast message from the client can reclaim the IP addresses that they offered to the client.

The DHCP server confirms that the IP address has been allocated to the client by returning a DHCPACK unicast message to the client. With this message, the client and server are bound, and the client uses configuration information received from the server. The amount of information the device receives depends on how you configure the DHCP server.

If the configuration parameters sent to the client in the DHCPOFFER unicast message are invalid (a configuration error exists), the client returns a DHCPDECLINE broadcast message to the DHCP server.

The DHCP server sends the client a DHCPNAK denial broadcast message, which means that the offered configuration parameters have not been assigned, that an error has occurred during the negotiation of the

parameters, or that the client has been slow in responding to the DHCPOFFER message (the DHCP server assigned the parameters to another client).

A DHCP client might receive offers from multiple DHCP or BOOTP servers and can accept any of the offers; however, the client usually accepts the first offer it receives. The offer from the DHCP server is not a guarantee that the IP address is allocated to the client; however, the server usually reserves the address until the client has had a chance to formally request the address. If the device accepts replies from a BOOTP server and configures itself, the device broadcasts, instead of unicasts, TFTP requests to obtain the device configuration file.

The DHCP hostname option allows a group of devices to obtain hostnames and a standard configuration from the central management DHCP server. A client (device) includes in its DCHPDISCOVER message an option 12 field used to request a hostname and other configuration parameters from the DHCP server. The configuration files on all clients are identical except for their DHCP-obtained hostnames.

DHCP-Based Autoconfiguration and Image Update

You can use the DHCP image upgrade features to configure a DHCP server to download both a new image and a new configuration file to one or more devices in a network. Simultaneous image and configuration upgrade for all switches in the network helps ensure that each new device added to a network receives the same image and configuration.

There are two types of DHCP image upgrades: DHCP autoconfiguration and DHCP auto-image update.

Restrictions for DHCP-Based Autoconfiguration

- The DHCP-based autoconfiguration with a saved configuration process stops if there is not at least one Layer 3 interface in an up state without an assigned IP address in the network.
- Unless you configure a timeout, the DHCP-based autoconfiguration with a saved configuration feature tries indefinitely to download an IP address.
- The auto-install process stops if a configuration file cannot be downloaded or if the configuration file is corrupted.
- The configuration file that is downloaded from TFTP is merged with the existing configuration in the running configuration but is not saved in the NVRAM unless you enter the **write memory** or **copy running-configuration startup-configuration** privileged EXEC command. If the downloaded configuration is saved to the startup configuration, the feature is not triggered during subsequent system restarts.

DHCP Autoconfiguration

DHCP autoconfiguration downloads a configuration file to one or more devices in your network from a DHCP server. The downloaded configuration file becomes the running configuration of the device. It does not over write the bootup configuration saved in the flash, until you reload the device.

DHCP Auto-Image Update

You can use DHCP auto-image upgrade with DHCP autoconfiguration to download both a configuration and a new image to one or more devices in your network. The devices (or devices) downloading the new configuration and the new image can be blank (or only have a default factory configuration loaded).

If the new configuration is downloaded to a switch that already has a configuration, the downloaded configuration is appended to the configuration file stored on the switch. (Any existing configuration is not overwritten by the downloaded one.)

To enable a DHCP auto-image update on the device, the TFTP server where the image and configuration files are located must be configured with the correct option 67 (the configuration filename), option 66 (the DHCP server hostname) option 150 (the TFTP server address), and option 125 (description of the Cisco IOS image file) settings.

After you install the device in your network, the auto-image update feature starts. The downloaded configuration file is saved in the running configuration of the device, and the new image is downloaded and installed on the device. When you reboot the device, the configuration is stored in the saved configuration on the device.

DHCP Server Configuration Guidelines

Follow these guidelines if you are configuring a device as a DHCP server:

- You should configure the DHCP server with reserved leases that are bound to each device by the device hardware address.
- If you want the device to receive IP address information, you must configure the DHCP server with these lease options:
  - IP address of the client (required)
  - Subnet mask of the client (required)
  - DNS server IP address (optional)
  - Router IP address (default gateway address to be used by the device) (required)
- If you want the device to receive the configuration file from a TFTP server, you must configure the DHCP server with these lease options:
  - TFTP server name (required)
  - Boot filename (the name of the configuration file that the client needs) (recommended)
  - Hostname (optional)
- Depending on the settings of the DHCP server, the device can receive IP address information, the configuration file, or both.
- If you do not configure the DHCP server with the lease options described previously, it replies to client requests with only those parameters that are configured. If the IP address and the subnet mask are not in the reply, the device is not configured. If the router IP address or the TFTP server name are not found, the device might send broadcast, instead of unicast, TFTP requests. Unavailability of other lease options does not affect autoconfiguration.

- The device can act as a DHCP server. By default, the Cisco IOS DHCP server and relay agent features are enabled on your device but are not configured. (These features are not operational.)

Purpose of the TFTP Server

Based on the DHCP server configuration, the device attempts to download one or more configuration files from the TFTP server. If you configured the DHCP server to respond to the device with all the options required for IP connectivity to the TFTP server, and if you configured the DHCP server with a TFTP server name, address, and configuration filename, the device attempts to download the specified configuration file from the specified TFTP server.

If you did not specify the configuration filename, the TFTP server, or if the configuration file could not be downloaded, the device attempts to download a configuration file by using various combinations of filenames and TFTP server addresses. The files include the specified configuration filename (if any) and these files: network-config, cisconet.cfg, *hostname*.config, or *hostname*.cfg, where *hostname* is the device's current hostname. The TFTP server addresses used include the specified TFTP server address (if any) and the broadcast address (255.255.255.255).

For the device to successfully download a configuration file, the TFTP server must contain one or more configuration files in its base directory. The files can include these files:

- The configuration file named in the DHCP reply (the actual device configuration file).
- The network-confg or the cisconet.cfg file (known as the default configuration files).
- The router-confg or the ciscortr.cfg file (These files contain commands common to all device. Normally, if the DHCP and TFTP servers are properly configured, these files are not accessed.)

If you specify the TFTP server name in the DHCP server-lease database, you must also configure the TFTP server name-to-IP-address mapping in the DNS-server database.

If the TFTP server to be used is on a different LAN from the device, or if it is to be accessed by the device through the broadcast address (which occurs if the DHCP server response does not contain all the required information described previously), a relay must be configured to forward the TFTP packets to the TFTP server. The preferred solution is to configure the DHCP server with all the required information.

Purpose of the DNS Server

The DHCP server uses the DNS server to resolve the TFTP server name to an IP address. You must configure the TFTP server name-to-IP address map on the DNS server. The TFTP server contains the configuration files for the device.

You can configure the IP addresses of the DNS servers in the lease database of the DHCP server from where the DHCP replies will retrieve them. You can enter up to two DNS server IP addresses in the lease database.

The DNS server can be on the same LAN or on a different LAN from the device. If it is on a different LAN, the device must be able to access it through a router.

How to Obtain Configuration Files

Depending on the availability of the IP address and the configuration filename in the DHCP reserved lease, the device obtains its configuration information in these ways:

- The IP address and the configuration filename is reserved for the device and provided in the DHCP reply (one-file read method).

  The device receives its IP address, subnet mask, TFTP server address, and the configuration filename from the DHCP server. The device sends a unicast message to the TFTP server to retrieve the named configuration file from the base directory of the server and upon receipt, it completes its boot up process.

- The IP address and the configuration filename is reserved for the device, but the TFTP server address is not provided in the DHCP reply (one-file read method).

  The device receives its IP address, subnet mask, and the configuration filename from the DHCP server. The device sends a broadcast message to a TFTP server to retrieve the named configuration file from the base directory of the server, and upon receipt, it completes its boot-up process.

- Only the IP address is reserved for the device and provided in the DHCP reply. The configuration filename is not provided (two-file read method).

  The device receives its IP address, subnet mask, and the TFTP server address from the DHCP server. The device sends a unicast message to the TFTP server to retrieve the network-confg or cisconet.cfg default configuration file. (If the network-confg file cannot be read, the device reads the cisconet.cfg file.)

  The default configuration file contains the hostnames-to-IP-address mapping for the device. The device fills its host table with the information in the file and obtains its hostname. If the hostname is not found in the file, the device uses the hostname in the DHCP reply. If the hostname is not specified in the DHCP reply, the device uses the default *Switch* as its hostname.

  After obtaining its hostname from the default configuration file or the DHCP reply, the device reads the configuration file that has the same name as its hostname (*hostname*-confg or *hostname*.cfg, depending on whether network-confg or cisconet.cfg was read earlier) from the TFTP server. If the cisconet.cfg file is read, the filename of the host is truncated to eight characters.

  If the device cannot read the network-confg, cisconet.cfg, or the hostname file, it reads the router-confg file. If the device cannot read the router-confg file, it reads the ciscortr.cfg file.

Note The device broadcasts TFTP server requests if the TFTP server is not obtained from the DHCP replies, if all attempts to read the configuration file through unicast transmissions fail, or if the TFTP server name cannot be resolved to an IP address.

Scheduled Reload of the Software Image

You can schedule a reload of the software image to occur on the device at a later time (for example, late at night or during the weekend when the device is used less), or you can synchronize a reload network-wide (for example, to perform a software upgrade on all device in the network).

Note A scheduled reload must take place within approximately 24 days.

You have these reload options:

- Reload of the software to take affect in the specified minutes or hours and minutes. The reload must take place within approximately 24 hours. You can specify the reason for the reload in a string up to 255 characters in length.
- Reload of the software to take place at the specified time (using a 24-hour clock). If you specify the month and day, the reload is scheduled to take place at the specified time and date. If you do not specify the month and day, the reload takes place at the specified time on the current day (if the specified time is later than the current time) or on the next day (if the specified time is earlier than the current time). Specifying 00:00 schedules the reload for midnight.

The **reload** command halts the system. If the system is not set to manually boot up, it reboots itself.

If your device is configured for manual booting, do not reload it from a virtual terminal. This restriction prevents the device from entering the boot loader mode and then taking it from the remote user's control.

If you modify your configuration file, the device prompts you to save the configuration before reloading. During the save operation, the system requests whether you want to proceed with the save if the CONFIG_FILE environment variable points to a startup configuration file that no longer exists. If you proceed in this situation, the system enters setup mode upon reload.

To cancel a previously scheduled reload, use the **reload cancel** privileged EXEC command.

How to Control Environment Variables

With a normally operating device, you enter the boot loader mode only through the console connection configured for 9600 bps.

The device boot loader software provides support for nonvolatile environment variables, which can be used to control how the boot loader, or any other software running on the system, operates. Boot loader environment variables are similar to environment variables that can be set on UNIX or DOS systems.

Environment variables that have values are stored in flash memory outside of the flash file system.

Each line in these files contains an environment variable name and an equal sign followed by the value of the variable. A variable has no value if it is not present; it has a value if it is listed even if the value is a null string. A variable that is set to a null string (for example, " ") is a variable with a value. Many environment variables are predefined and have default values.

You can change the settings of the environment variables by accessing the boot loader or by using Cisco IOS commands. Under normal circumstances, it is not necessary to alter the setting of the environment variables.

Common Environment Variables

This table describes the function of the most common environment variables.

Table 9: Common Environment Variables

Variable	Boot Loader Command	Cisco IOS Global Configuration Command
BOOT	**set BOOT** *filesystem* **:/** *file-url* ... A semicolon-separated list of executable files to try to load and execute when automatically booting.	**boot system** {*filesystem* **:** */file-url* ... | **switch** {*number* | **all**}} Specifies the Cisco IOS image to load during the next boot cycle . This command changes the setting of the BOOT environment variable. The package provisioning file, also referred to as the *packages.conf* file, is used by the system to determine which software packages to activate during boot up. • When booting in installed mode, the package provisioning file specified in the **boot** command is used to determine which packages to activate. For example **boot flash:packages.conf**. • When booting in bundle mode, the package provisioning file contained in the booted bundle is used to activate the packages included in the bundle. For example, **boot flash:image.bin**.
MANUAL_BOOT	**set MANUAL_BOOT yes** Decides whether the switch automatically or manually boots. Valid values are 1, yes, 0, and no. If it is set to no or 0, the boot loader attempts to automatically boot up the system. If it is set to anything else, you must manually boot up the switch from the boot loader mode.	**boot manual** Enables manually booting the switch during the next boot cycle and changes the setting of the MANUAL_BOOT environment variable. The next time you reboot the system, the switch is in boot loader mode. To boot up the system, use the **boot flash:** *filesystem* **:/** *file-url* boot loader command, and specify the name of the bootable image.

Variable	Boot Loader Command	Cisco IOS Global Configuration Command
CONFIG_FILE	**set CONFIG_FILE flash:/** *file-url* Changes the filename that Cisco IOS uses to read and write a nonvolatile copy of the system configuration.	**boot config-file flash:/** *file-url* Specifies the filename that Cisco IOS uses to read and write a nonvolatile copy of the system configuration. This command changes the CONFIG_FILE environment variable.
BAUD	**set BAUD** *baud-rate*	**line console 0** **speed** *speed-value* Configures the baud rate.
ENABLE_BREAK	**set ENABLE_BREAK yes/no**	**boot enable-break switch yes/no** Enables a break to the auto-boot cycle. You have 5 seconds to enter the **break** command.

Environment Variables for TFTP

When the switch is connected to a PC through the Ethernet management port, you can download or upload a configuration file to the boot loader by using TFTP. Make sure the environment variables in this table are configured.

Table 10: Environment Variables for TFTP

Variable	Description
MAC_ADDR	Specifies the MAC address of the switch. **Note** We recommend that you do not modify this variable. However, if you modify this variable after the boot loader is up or the value is different from the saved value, enter this command before using TFTP. A reset is required for the new value to take effect.
IP_ADDRESS	Specifies the IP address and the subnet mask for the associated IP subnet of the switch.
DEFAULT_GATEWAY	Specifies the IP address and subnet mask of the default gateway.

How to Perform Device Setup Configuration

Using DHCP to download a new image and a new configuration to a device requires that you configure at least two devices. One device acts as a DHCP and TFTP server and the second device (client) is configured to download either a new configuration file or a new configuration file and a new image file.

Configuring DHCP Autoconfiguration (Only Configuration File)

This task describes how to configure DHCP autoconfiguration of the TFTP and DHCP settings on an existing device in the network so that it can support the autoconfiguration of a new device.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**ip dhcp pool** *poolname* **Example:** `Device(config)# ip dhcp pool pool`	Creates a name for the DHCP server address pool, and enters DHCP pool configuration mode.
Step 3	**boot** *filename* **Example:** `Device(dhcp-config)# boot config-boot.text`	Specifies the name of the configuration file that is used as a boot image.
Step 4	**network** *network-number mask prefix-length* **Example:** `Device(dhcp-config)# network 10.10.10.0 255.255.255.0`	Specifies the subnet network number and mask of the DHCP address pool. **Note** The prefix length specifies the number of bits that comprise the address prefix. The prefix is an alternative way of specifying the network mask of the client. The prefix length must be preceded by a forward slash (/).
Step 5	**default-router** *address* **Example:** `Device(dhcp-config)# default-router 10.10.10.1`	Specifies the IP address of the default router for a DHCP client.
Step 6	**option 150** *address* **Example:** `Device(dhcp-config)# option 150 10.10.10.1`	Specifies the IP address of the TFTP server.
Step 7	**exit** **Example:**	Returns to global configuration mode.

	Command or Action	Purpose
	`Device(dhcp-config)# exit`	
Step 8	**tftp-server flash**:*filename.text* **Example:** `Device(config)# tftp-server flash:config-boot.text`	Specifies the configuration file on the TFTP server.
Step 9	**interface** *interface-id* **Example:**	Specifies the address of the client that will receive the configuration file.
Step 10	**no switchport** **Example:** `Device(config-if)# no switchport`	Puts the interface into Layer 3 mode.
Step 11	**ip address** *address mask* **Example:** `Device(config-if)# ip address 10.10.10.1 255.255.255.0`	Specifies the IP address and mask for the interface.
Step 12	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring DHCP Auto-Image Update (Configuration File and Image)

This task describes DHCP autoconfiguration to configure TFTP and DHCP settings on an existing device to support the installation of a new switch.

Before you begin

You must first create a text file (for example, autoinstall_dhcp) that will be uploaded to the device. In the text file, put the name of the image that you want to download (for example, ess9300_iosxe.17.xx.xx.SPA.bin).

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 2	**ip dhcp pool** *poolname* **Example:** `Device(config)# ip dhcp pool pool1`	Creates a name for the DHCP server address pool and enter DHCP pool configuration mode.
Step 3	**boot** *filename* **Example:** `Device(dhcp-config)# boot config-boot.text`	Specifies the name of the file that is used as a boot image.
Step 4	**network** *network-number mask prefix-length* **Example:** `Device(dhcp-config)# network 10.10.10.0 255.255.255.0`	Specifies the subnet network number and mask of the DHCP address pool. **Note** The prefix length specifies the number of bits that comprise the address prefix. The prefix is an alternative way of specifying the network mask of the client. The prefix length must be preceded by a forward slash (/).
Step 5	**default-router** *address* **Example:** `Device(dhcp-config)# default-router 10.10.10.1`	Specifies the IP address of the default router for a DHCP client.
Step 6	**option 150** *address* **Example:** `Device(dhcp-config)# option 150 10.10.10.1`	Specifies the IP address of the TFTP server.
Step 7	**option 125** *hex* **Example:** `Device(dhcp-config)# option 125 hex 0000.0009.0a05.08661.7574.6f69.6e73.7461.6c6c.5f64.686370`	Specifies the path to the text file that describes the path to the image file.
Step 8	**copy tftp flash** *filename.txt* **Example:** `Device(config)# copy tftp flash image.bin`	Uploads the text file to the device.

	Command or Action	Purpose
Step 9	**copy tftp flash** *imagename.bin* **Example:** `Device(config)# copy tftp flash image.bin`	Uploads the tar file for the new image to the device.
Step 10	**exit** **Example:** `Device(dhcp-config)# exit`	Returns to global configuration mode.
Step 11	**tftp-server flash:** *config.text* **Example:** `Device(config)# tftp-server flash:config-boot.text`	Specifies the Cisco IOS configuration file on the TFTP server.
Step 12	**tftp-server flash:** *imagename.bin* **Example:** `Device(config)# tftp-server flash:image.bin`	Specifies the image name on the TFTP server.
Step 13	**tftp-server flash:** *filename.txt* **Example:** `Device(config)# tftp-server flash:boot-config.text`	Specifies the text file that contains the name of the image file to download
Step 14	**interface** *interface-id* **Example:** `Device(config)# interface gigabitEthernet1/0/4`	Specifies the address of the client that will receive the configuration file.
Step 15	**no switchport** **Example:** `Device(config-if)# no switchport`	Puts the interface into Layer 3 mode.
Step 16	**ip address** *address mask* **Example:** `Device(config-if)# ip address 10.10.10.1`	Specifies the IP address and mask for the interface.

	Command or Action	Purpose
	`255.255.255.0`	
Step 17	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 18	**copy running-config startup-config** **Example:** `Device(config-if)# end`	(Optional) Saves your entries in the configuration file.

Configuring the Client to Download Files from DHCP Server

Note You should only configure and enable the Layer 3 interface. Do not assign an IP address or DHCP-based autoconfiguration with a saved configuration.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**boot host dhcp** **Example:** `Device(conf)# boot host dhcp`	Enables autoconfiguration with a saved configuration.
Step 3	**boot host retry timeout** *timeout-value* **Example:** `Device(conf)# boot host retry timeout 300`	(Optional) Sets the amount of time the system tries to download a configuration file. **Note** If you do not set a timeout, the system will try indefinitely to obtain an IP address from the DHCP server.
Step 4	**banner config-save ^C** *warning-message* **^C** **Example:**	(Optional) Creates warning messages to be displayed when you try to save the configuration file to NVRAM.

	Command or Action	Purpose
	`Device(conf)# banner config-save ^C Caution - Saving Configuration File to NVRAM May Cause You to No longer Automatically Download Configuration Files at Reboot^C`	
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show boot** **Example:** `Device# show boot`	Verifies the configuration.

Manually Assigning IP Information to Multiple SVIs

This task describes how to manually assign IP information to multiple switched virtual interfaces (SVIs):

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface vlan** *vlan-id* **Example:** `Device(config)# interface vlan 99`	Enters interface configuration mode, and enters the VLAN to which the IP information is assigned. The range is 1 to 4094.
Step 4	**ip address** *ip-address subnet-mask* **Example:** `Device(config-vlan)# ip address 10.10.10.2`	Enters the IP address and subnet mask.

	Command or Action	Purpose
	`255.255.255.0`	
Step 5	**exit** **Example:** `Device(config-vlan)# exit`	Returns to global configuration mode.
Step 6	**ip default-gateway** *ip-address* **Example:** `Device(config)# ip default-gateway 10.10.10.1`	Enters the IP address of the next-hop router interface that is directly connected to the device where a default gateway is being configured. The default gateway receives IP packets with unresolved destination IP addresses from the device. Once the default gateway is configured, the device has connectivity to the remote networks with which a host needs to communicate. **Note** When your device is configured to route with IP, it does not need to have a default gateway set. **Note** The device capwap relays on default-gateway configuration to support routed access point join the device.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show interfaces vlan** *vlan-id* **Example:** `Device# show interfaces vlan 99`	Displays the interfaces status for the specified VLAN.
Step 9	**show ip redirects** **Example:** `Device# show ip redirects`	Displays the Internet Control Message Protocol (ICMP) redirect messages.

Modifying Device Startup Configuration

The following sections provide information on how to modify the startup configuration of a device.

Specifying a Filename to Read and Write a System Configuration

By default, the Cisco IOS software uses the config.text file to read and write a nonvolatile copy of the system configuration. However, you can specify a different filename, which will be loaded during the next boot cycle.

Before you begin

Use a standalone device for this task.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**boot flash**:*/file-url* **Example:** `Device(config)# boot flash:config.text`	Specifies the configuration file to load during the next boot cycle. • *file-url*: The path (directory) and the configuration filename. • Filenames and directory names are case-sensitive.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show boot** **Example:** `Device# show boot`	Lists the contents of the BOOT environment variable (if set), the name of the configuration file pointed to by the CONFIG_FILE environment variable, and the contents of the BOOTLDR environment variable. • The **boot** global configuration command changes the setting of the CONFIG_FILE environment variable.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Scheduled Software Image Reload

This task describes how to configure your device to reload the software image at a later time.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves your device configuration information to the startup configuration before you use the **reload** command.

Step 4	**reload in** [*hh*:]*mm* [*text*]
	Example:
	`Device# reload in 12`
	`System configuration has been modified. Save? [yes/no]: y`
		Schedules a reload of the software to take affect in the specified minutes or hours and minutes. The reload must take place within approximately 24 days. You can specify the reason for the reload in a string up to 255 characters in length.

Step 5	**reload at** *hh*: *mm* [*month day* | *day month*] [*text*] **Example:** `Device(config)# reload at 14:00`	Specifies the time in hours and minutes for the reload to occur. **Note** Use the **at** keyword only if the device system clock has been set (through Network Time Protocol (NTP), the hardware calendar, or manually). The time is relative to the configured time zone on the device. To schedule reloads across several devices to occur simultaneously, the time on each device must be synchronized with NTP.
Step 6	**reload cancel** **Example:** `Device(config)# reload cancel`	Cancels a previously scheduled reload.

	Command or Action	Purpose
Step 7	**show reload** **Example:** `show reload`	Displays information about a previously scheduled reload or identifies if a reload has been scheduled on the device.

Configuration Examples for Device Setup Configuration

The following sections provide configuration examples for device setup.

Examples: Displaying Software Bootup in Install Mode

The following example displays software bootup in install mode:

```
switch: boot flash:packages.conf
Attempting to boot from [flash:packages.conf]
Located packages.conf
#

validate_package: SHA-1 hash:
        expected 340D5091:2872A0DD:03E9068C:3FDBECAB:69786462
        calculated 340D5091:2872A0DD:03E9068C:3FDBECAB:69786462
Image parsed from conf file is cat9k-rpboot.16.09.01.SPA.pkg
################################################################################################################


Waiting for 120 seconds for other switches to boot
################################################################################################################
Switch number is 1

              Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

           cisco Systems, Inc.
           170 West Tasman Drive
           San Jose, California 95134-1706



Cisco IOS Software [Fuji], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 16.9.1, RELEASE
 SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2017 by Cisco Systems, Inc.
Compiled Tue 30-May-17 00:36 by mcpre



Cisco IOS-XE software, Copyright (c) 2005-2017 by cisco Systems, Inc.
All rights reserved.  Certain components of Cisco IOS-XE software are
licensed under the GNU General Public License ("GPL") Version 2.0.  The
software code licensed under GPL Version 2.0 is free software that comes
```

```
with ABSOLUTELY NO WARRANTY.  You can redistribute and/or modify such
GPL code under the terms of GPL Version 2.0.  For more details, see the
documentation or "License Notice" file accompanying the IOS-XE software,
or the applicable URL provided on the flyer accompanying the IOS-XE
software.



FIPS: Flash Key Check : Begin
FIPS: Flash Key Check : End, Not Found, FIPS Mode Not Enabled

This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply
third-party authority to import, export, distribute or use encryption.
Importers, exporters, distributors and users are responsible for
compliance with U.S. and local country laws. By using this product you
agree to comply with applicable laws and regulations. If you are unable
to comply with U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html

If you require further assistance please contact us by sending email to
export@cisco.com.
cisco C9200L-24P-4G (ARM64) processor with 518473K/3071K bytes of memory.
Processor board ID JPG221000RH
988 Virtual Ethernet interfaces
56 Gigabit Ethernet interfaces
2048K bytes of non-volatile configuration memory.
2015456K bytes of physical memory.
819200K bytes of Crash Files at crashinfo:.
1941504K bytes of Flash at flash:.
0K bytes of WebUI ODM Files at webui:.
819200K bytes of Crash Files at crashinfo-7:.
1941504K bytes of Flash at flash-7:.

Base Ethernet MAC Address          : 68:2c:7b:f7:49:00
Motherboard Assembly Number        : 73-18699-2
Motherboard Serial Number          : JAE22090AZB
Model Revision Number              : 13
Motherboard Revision Number        : 05
Model Number                       : C9200L-24P-4G
System Serial Number               : JPG221000RH


%INIT: waited 0 seconds for NVRAM to be available

Defaulting CPP : Policer rate for all classes will be set to their defaults


Press RETURN to get started!
```

The following example displays software bootup in bundle mode:

```
switch: boot flash: cat9k_lite_iosxe.16.09.01.SPA.bin

Attempting to boot from [flash: cat9k_lite_iosxe.16.09.01.SPA.bin]
Located cat9k_lite_iosxe.16.09.01.SPA.bin
############################################################################################################################
Warning: ignoring ROMMON var "BOOT_PARAM"
```

```
Waiting for 120 seconds for other switches to boot
############################################################################################################################
Switch number is 3

              Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

           cisco Systems, Inc.
           170 West Tasman Drive
           San Jose, California 95134-1706



Cisco IOS Software [Fuji], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 16.9.1, RELEASE
 SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2017 by Cisco Systems, Inc.
Compiled Tue 30-May-17 00:36 by mcpre



Cisco IOS-XE software, Copyright (c) 2005-2017 by cisco Systems, Inc.
All rights reserved.  Certain components of Cisco IOS-XE software are
licensed under the GNU General Public License ("GPL") Version 2.0.  The
software code licensed under GPL Version 2.0 is free software that comes
with ABSOLUTELY NO WARRANTY.  You can redistribute and/or modify such
GPL code under the terms of GPL Version 2.0.  For more details, see the
documentation or "License Notice" file accompanying the IOS-XE software,
or the applicable URL provided on the flyer accompanying the IOS-XE
software.



FIPS: Flash Key Check : Begin
FIPS: Flash Key Check : End, Not Found, FIPS Mode Not Enabled

This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply
third-party authority to import, export, distribute or use encryption.
Importers, exporters, distributors and users are responsible for
compliance with U.S. and local country laws. By using this product you
agree to comply with applicable laws and regulations. If you are unable
to comply with U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html

If you require further assistance please contact us by sending email to
export@cisco.com.

cisco C9200L-24P-4G (ARM64) processor with 518473K/3071K bytes of memory.
Processor board ID JPG221000RH
988 Virtual Ethernet interfaces
56 Gigabit Ethernet interfaces
2048K bytes of non-volatile configuration memory.
```

```
2015456K bytes of physical memory.
819200K bytes of Crash Files at crashinfo:.
1941504K bytes of Flash at flash:.
0K bytes of WebUI ODM Files at webui:.
819200K bytes of Crash Files at crashinfo-7:.
1941504K bytes of Flash at flash-7:.

Base Ethernet MAC Address          : 68:2c:7b:f7:49:00
Motherboard Assembly Number        : 73-18699-2
Motherboard Serial Number          : JAE22090AZB
Model Revision Number              : 13
Motherboard Revision Number        : 05
Model Number                       : C9200L-24P-4G
System Serial Number               : JPG221000RH


%INIT: waited 0 seconds for NVRAM to be available

Defaulting CPP : Policer rate for all classes will be set to their defaults


Press RETURN to get started!
```

Example: Emergency Installation

The following is a sample output of the **emergency-install** boot command:

```
switch: emergency-install tftp://10.10.0.10/auto/tftpboot/X86/cat9k_iosxe.17.04.01.SPA.bin
WARNING: The system partition (bootflash:) will be erased during the system recovery install
 process.
Are you sure you want to proceed? [y]  y/n [n]: y
Starting system recovery (tftp://210.10.0.10/auto/tftpboot/X86/cat9k_iosxe.17.04.01.SPA.bin)
 ...
Attempting to boot from [sda9:cat9k-recovery.SSA.bin]
Located cat9k-recovery.SSA.bin
############################################################################################################################################

Warning: ignoring ROMMON var "BOOT_PARAM"

PLATFORM_TYPE 9300 speed 9600

Booting Recovery Image 17.4.1



Initiating Emergency Installation of bundle
tftp://10.10.0.10/auto/tftpboot/X86/cat9k_iosxe.17.04.01.SPA.bin


Downloading bundle tftp://10.10.0.10/auto/tftpboot/X86/cat9k_iosxe.17.04.01.SPA.bin...
curl_vrf=2
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  485M  100  485M    0     0  5143k      0  0:01:36  0:01:36 --:--:-- 5256k
100  485M  100  485M    0     0  5143k      0  0:01:36  0:01:36 --:--:-- 5143k

Validating bundle tftp://10.10.0.10/auto/tftpboot/X86/cat9k_iosxe.17.04.01.SPA.bin...
Installing bundle tftp://10.10.0.10/auto/tftpboot/X86/cat9k_iosxe.17.04.01.SPA.bin....
Verifying bundle tftp://10.10.0.10/auto/tftpboot/X86/cat9k_iosxe.17.04.01.SPA.bin...
Package cat9k-cc_srdriver.17.04.01.SPA.pkg /temp//stage/cat9k-cc_srdriver.17.04.01.SPA.pkg
 is Digitally Signed
```

```
Package cat9k-espbase.17.04.01.SPA.pkg /temp//stage/cat9k-espbase.17.04.01.SPA.pkg is
Digitally Signed
Package cat9k-guestshell.17.04.01.SPA.pkg /temp//stage/cat9k-guestshell.17.04.01.SPA.pkg
is Digitally Signed
Package cat9k-rpbase.17.04.01.SPA.pkg /temp//stage/cat9k-rpbase.17.04.01.SPA.pkg is Digitally
 Signed
Package cat9k-sipbase.17.04.01.SPA.pkg /temp//stage/cat9k-sipbase.17.04.01.SPA.pkg is
Digitally Signed
Package cat9k-sipspa.17.04.01.SPA.pkg /temp//stage/cat9k-sipspa.17.04.01.SPA.pkg is Digitally
 Signed
Package cat9k-srdriver.17.04.01.SPA.pkg /temp//stage/cat9k-srdriver.17.04.01.SPA.pkg is
Digitally Signed
Package cat9k-webui.17.04.01.SPA.pkg /temp//stage/cat9k-webui.17.04.01.SPA.pkg is Digitally
 Signed
Package cat9k-wlc.17.04.01.SPA.pkg /temp//stage/cat9k-wlc.17.04.01.SPA.pkg is Digitally
Signed
Package /cat9k-rpboot.17.04.01.SPA.pkg /temp//rpboot/cat9k-rpboot.17.04.01.SPA.pkg is
Digitally Signed
Preparing flash....
Flash filesystem unmounted successfully /dev/sdb3
Syncing device....
Emergency Install successful... Rebooting
Will reboot now

Initializing Hardware...

System Bootstrap, Version 17.04.01, RELEASE SOFTWARE (P)
Compiled Wed 05/31/2020 15:58:35.22 by rel

Current image running:
Primary Rommon Image

Last reset cause: Software Reload
```

Example: Managing an Update Package

The following example shows how to add a software package file:

```
Device# install add file flash:cat9k_lite_iosxe.17.04.01.SPA.bin activate commit

install_add_activate_commit: START Thu Aug 30 20:25:35 IST 2018

Aug 30 20:25:38.688 IST: %INSTALL-5-INSTALL_START_INFO: Switch 7 R0/0: install_engine:
Started install one-shot flash:cat9k_lite_iosxe.17.04.01.SPA.bininstall_add_activate_commit:
 Adding PACKAGE

This operation requires a reload of the system. Do you want to proceed?
Please confirm you have changed boot config to flash:packages.conf [y/n]y

--- Starting initial file syncing ---
[7]: Copying flash:cat9k_lite_iosxe.17.04.01.SPA.bin from switch 7 to switch 4
[4]: Finished copying to switch 4
Info: Finished copying flash:cat9k_lite_iosxe.17.04.01.SPA.bin to the selected switch(es)
Finished initial file syncing

--- Starting Add ---
Performing Add on all members
  [4] Add package(s) on switch 4
  [4] Finished Add on switch 4
  [7] Add package(s) on switch 7
  [7] Finished Add on switch 7
```

```
Checking status of Add on [4 7]
Add: Passed on [4 7]
Finished Add

install_add_activate_commit: Activating PACKAGE

gzip: initramfs.cpio.gz: decompression OK, trailing garbage ignored
Following packages shall be activated:
/flash/cat9k_lite-webui.17.04.01.SPA.pkg
/flash/cat9k_lite-srdriver.17.04.01.SPA.pkg
/flash/cat9k_lite-rpboot.17.04.01.SPA.pkg
/flash/cat9k_lite-rpbase.17.04.01.SPA.pkg

This operation requires a reload of the system. Do you want to proceed? [y/n]y
--- Starting Activate ---
Performing Activate on all members

Aug 30 20:51:16.365 IST: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 7 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds  [4] Activate package(s)
 on switch 4
  [4] Finished Activate on switch 4
  [7] Activate package(s) on switch 7

Aug 30 20:51:17.561 IST: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 4 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds  [7] Finished Activate
 on switch 7
Checking status of Activate on [4 7]
Activate: Passed on [4 7]
Finished Activate

--- Starting Commit ---
Performing Commit on all members
  [4] Commit package(s) on switch 4
  [4] Finished Commit on switch 4
  [7] Commit package(s) on switch 7
  [7] Finished Commit on switch 7
Checking status of Commit on [4 7]
Commit: Passed on [4 7]
Finished Commit

Install will reload the system now!
SUCCESS: install_add_activate_commit  Thu Aug 30 20:51:55 IST 2018

Y2#
   Chassis 7 reloading, reason - Reload command

Aug 30 20:51:56.017 IST: %INSTALL-5-INSTALL_COMPLETED_INFO: Switch 7 R0/0: install_engine:
 Completed install one-shot PACKAGE flash:cat9k_lite_iosxe.17.04.01.SPA.binAug 30
20:52:03.517: %PMAN-5-EXITACTION: F0/0: pvp: Process manager is exiting: reload fp action
requested
Aug 30 20:52:07.543: %PMAN-5-EXITACTION: R0/0: pvp: Process manager is exiting: rp processes
 exit with reload switch code

Aug 30 20:52:11.104: %PMAN-5-EXITACTION: C0/0: pvp: Process manager is exiting: reload cc
action requested
reboot: Restarting system
```

The following is a sample output of the **show install summary** command after adding a software package file to a device:

```
Device# show install summary
[ Switch 4 7 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
```

```
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
IMG   C    16.9.1.0.70

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

The following example shows how to activate an added software package file:

```
Device# install activate

install_activate: START Mon Oct 30 20:14:20 UTC 2017
install_activate: Activating PACKAGE

*Oct 30 20:14:21.379: %IOSXE-5-PLATFORM: Switch 1 R0/0: Oct 30 20:14:21 install_engine.sh:

%INSTALL-5-INSTALL_START_INFO: Started install activateFollowing packages shall be activated:
/flash/cat9k-wlc.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
/flash/cat9k-webui.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
/flash/cat9k-srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
/flash/cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
/flash/cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
/flash/cat9k-rpboot.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
/flash/cat9k-rpbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
/flash/cat9k-guestshell.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
/flash/cat9k-espbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
/flash/cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg

This operation requires a reload of the system. Do you want to proceed? [y/n]y
--- Starting Activate ---
Performing Activate on all members
  [1] Activate package(s) on switch 1
    --- Starting list of software package changes ---
    Old files list:
      Removed cat9k-cc_srdriver.16.06.02.SPA.pkg
      Removed cat9k-espbase.16.06.02.SPA.pkg
      Removed cat9k-guestshell.16.06.02.SPA.pkg
      Removed cat9k-rpbase.16.06.02.SPA.pkg
      Removed cat9k-rpboot.16.06.02.SPA.pkg
      Removed cat9k-sipbase.16.06.02.SPA.pkg
      Removed cat9k-sipspa.16.06.02.SPA.pkg
      Removed cat9k-srdriver.16.06.02.SPA.pkg
      Removed cat9k-webui.16.06.02.SPA.pkg
      Removed cat9k-wlc.16.06.02.SPA.pkg
    New files list:
      Added cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
      Added cat9k-espbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
      Added cat9k-guestshell.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
      Added cat9k-rpbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
      Added cat9k-rpboot.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
      Added cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
      Added cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
      Added cat9k-srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
      Added cat9k-webui.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
      Added cat9k-wlc.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Finished list of software package changes
  [1] Finished Activate on switch 1
Checking status of Activate on [1]
Activate: Passed on [1]
Finished Activate
```

```
*Oct 30 20:15:56.572: %IOSXE-5-PLATFORM: Switch 1 R0/0: Oct 30 20:15:56 rollback_timer.sh:
 %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Install auto abort timer will expire in 7200
 seconds
 Install will reload the system now!
SUCCESS: install_activate  Mon Oct 30 20:16:01 UTC 2017

Device#
*Oct 30 20:16:01.935: %IOSXE-5-PLATFORM: Switch 1 R0/0: Oct 30 20:16:01
 install_engine.sh: %INSTALL-5-INSTALL_COMPLETED_INFO: Completed install activate PACKAGE
                                    Chassis 1 reloading, reason - Reload command
```

The following sample output from the **show install summary** command displays the status of the software package as active and uncommitted:

```
Device# show install summary

[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
IMG   I    16.6.2.0
IMG   U    16.6.1.0
Device#
```

The following example shows how to execute the **install commit** command:

The following example shows how to rollback an update package to the base package:

The following is a sample output from the **install remove inactive** command:

The following is sample output from the **install abort** command:

The following is a sample output from the **install activate auto-abort-timer** command:

Verifying Software Install

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **show install log**

Example:

```
Device# show install log
```

Displays information about all the software install operations that was performed since boot-up of the device.

```
Device# show install log
[0|install_op_boot]: START Tue Aug 30 06:39:48 Universal 2018
[0|install_op_boot]: END SUCCESS  Tue Aug 30 06:39:50 Universal 2018
```

Step 3 **show install summary**

Example:

```
Device# show install summary
```

Displays information about the image versions and their corresponding install state for all members/field-replaceable unit (FRU).

- The output of this command differs based on the **install** command that is executed.

```
Device# show install summary
[ Switch 1 2 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
IMG   C    16.9.1.0.70

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

Step 4 **show install package** *filesystem: filename*

Example:

```
Device# show install package flash:cat9k_lite-rpboot.17.04.01.SPA.pkg
```

Displays information about the specified software install package file.

```
Device# show install package flash:cat9k_lite-rpboot.17.04.01.SPA.pkg
Package: cat9k_lite-rpboot.17.04.01.SPA.pkg
  Size: 34616705
  Timestamp: Thu Aug 30 20:28:25 2018 UTC
  Canonical path: /flash/cat9k_lite-rpboot.17.04.01.SPA.pkg

  Raw disk-file SHA1sum:
    5e816f97bcae3e30eb8bc2f0ec8f64402cea1638
  Header size:     980 bytes
  Package type:    30001
  Package flags:   0
  Header version:  3

  Package is bootable on RP when specified
  by packages provisioning file.
```

Example: Configuring a Device to Download Configurations from a DHCP Server

The following example shows how to use a Layer 3 SVI interface on VLAN 99 to enable DHCP-based autoconfiguration with a saved configuration:

```
Device# configure terminal
Device(config)# boot host dhcp
Device(config)# boot host retry timeout 300
Device(config)# banner config-save ^C Caution - Saving Configuration File to NVRAM May Cause
 You to No longer Automatically Download Configuration Files at Reboot^C
Device(config)# vlan 99
Device(config-vlan)# interface vlan 99
Device(config-if)# no shutdown
Device(config-if)# end
Device# show boot

BOOT path-list:
Config file:          flash:/config.text
Private Config file:  flash:/private-config.text
Enable Break:         no
Manual Boot:          no
HELPER path-list:
NVRAM/Config file
      buffer size:    32768
Timeout for Config
          Download:    300 seconds
Config Download
       via DHCP:       enabled (next boot: enabled)
Device#
```

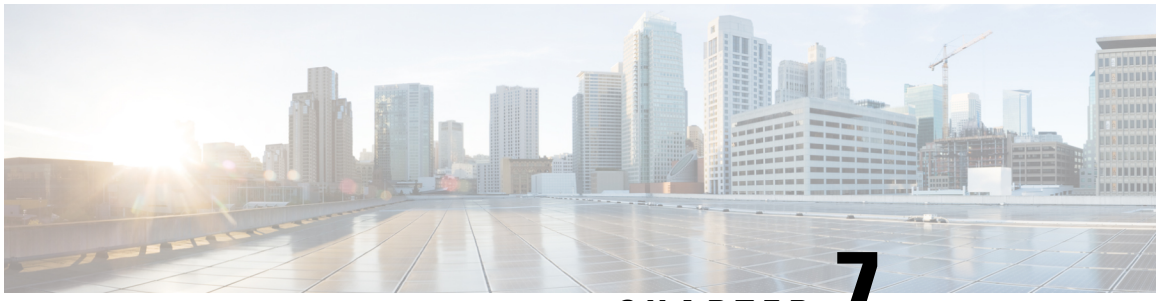
Example: Scheduling Software Image Reload

This example shows how to reload the software on a device on the current day at 7:30 p.m:

```
Device# reload at 19:30

Reload scheduled for 19:30:00 UTC Wed Jun 5 2013 (in 2 hours and 25 minutes)
Proceed with reload? [confirm]
```

This example shows how to reload the software on a device at a future date and time:

```
Device# reload at 02:00 jun 20

Reload scheduled for 02:00:00 UTC Thu Jun 20 2013 (in 344 hours and 53 minutes)
Proceed with reload? [confirm]
```



CHAPTER 7

Configuring System Message Logs

Information About Configuring System Message Logs

System Messsage Logging

By default, a switch sends the output from system messages and **debug** privileged EXEC commands to a logging process. . The logging process controls the distribution of logging messages to various destinations, such as the logging buffer, terminal lines, or a UNIX syslog server, depending on your configuration. The process also sends messages to the console.

When the logging process is disabled, messages are sent only to the console. The messages are sent as they are generated, so message and debug output are interspersed with prompts or output from other commands. Messages appear on the active consoles after the process that generated them has finished.

You can set the severity level of the messages to control the type of messages displayed on the consoles and each of the destinations. You can time-stamp log messages or set the syslog source address to enhance real-time debugging and management. For information on possible messages, see the system message guide for this release.

You can access logged system messages by using the switch command-line interface (CLI) or by saving them to a properly configured syslog server. The switch software saves syslog messages in an internal buffer on a standalone switch. If a standalone switch , the log is lost unless you had saved it to flash memory.

You can remotely monitor system messages by viewing the logs on a syslog server or by accessing the switch through Telnet, through the console port, or through the Ethernet management port.

Note The syslog format is compatible with 4.3 BSD UNIX.

System Log Message Format

System log messages can contain up to 80 characters and a percent sign (%), which follows the optional sequence number or time-stamp information, if configured. Depending on the switch, messages appear in one of these formats:

- *seq no:timestamp: %facility-severity-MNEMONIC:description (hostname-n)*
- *seq no:timestamp: %facility-severity-MNEMONIC:description*

The part of the message preceding the percent sign depends on the setting of these global configuration commands:

- **service sequence-numbers**
- **service timestamps log datetime**
- **service timestamps log datetime** [**localtime**] [**msec**] [**show-timezone**]
- **service timestamps log uptime**

Table 11: System Log Message Elements

Element	Description
seq no:	Stamps log messages with a sequence number only if the **service sequence-numbers** global configuration command is configured.
timestamp formats: *mm/dd h h:mm:ss* or *hh:mm:ss* (short uptime) or *d h* (long uptime)	Date and time of the message or event. This information appears only if the **service timestamps log** [**datetime** | **log**] global configuration command is configured.
facility	The facility to which the message refers (for example, SNMP, SYS, and so forth).
severity	Single-digit code from 0 to 7 that is the severity of the message.
MNEMONIC	Text string that uniquely describes the message.
description	Text string containing detailed information about the event being reported.

Default System Message Logging Settings

Table 12: Default System Message Logging Settings

Feature	Default Setting
System message logging to the console	Enabled.

Feature	Default Setting
Console severity	Debugging.
Logging file configuration	No filename specified.
Logging buffer size	4096 bytes.
Logging history size	1 message.
Time stamps	Disabled.
Synchronous logging	Disabled.
Logging server	Disabled.
Syslog server IP address	None configured.
Server facility	Local7
Server severity	Informational.

Syslog Message Limits

If you enabled syslog message traps to be sent to an SNMP network management station by using the **snmp-server enable trap** global configuration command, you can change the level of messages sent and stored in the switch history table. You also can change the number of messages that are stored in the history table.

Messages are stored in the history table because SNMP traps are not guaranteed to reach their destination. By default, one message of the level **warning** and numerically lower levels are stored in the history table even if syslog traps are not enabled.

When the history table is full (it contains the maximum number of message entries specified with the **logging history size** global configuration command), the oldest message entry is deleted from the table to allow the new message entry to be stored.

The history table lists the level keywords and severity level. For SNMP usage, the severity level values increase by 1. For example, *emergencies* equal 1, not 0, and *critical* equals 3, not 2.

How to Configure System Message Logs

Setting the Message Display Destination Device

If message logging is enabled, you can send messages to specific locations in addition to the console.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**logging buffered** [*size*] **Example:** `Device(config)# logging buffered 8192`	Logs messages to an internal buffer on the switch. The range is 4096 to 2147483647 bytes. The default buffer size is 4096 bytes. If a standalone switchfails, the log file is lost unless you previously saved it to flash memory. See Step 4. **Note** Do not make the buffer size too large because the switch could run out of memory for other tasks. Use the **show memory** privileged EXEC command to view the free processor memory on the switch. However, this value is the maximum available, and the buffer size should *not* be set to this amount.
Step 3	**logging** *host* **Example:** `Device(config)# logging 125.1.1.100`	Logs messages to a UNIX syslog server host. *host* specifies the name or IP address of the host to be used as the syslog server. To build a list of syslog servers that receive logging messages, enter this command more than once.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**terminal monitor** **Example:** `Device# terminal monitor`	Logs messages to a nonconsole terminal during the current session. Terminal parameter-setting commands are set locally and do not remain in effect after the session has ended. You must perform this step for each session to see the debugging messages.

Synchronizing Log Messages

You can synchronize unsolicited messages and **debug** privileged EXEC command output with solicited device output and prompts for a specific console port line or virtual terminal line. You can identify the types of messages to be output asynchronously based on the level of severity. You can also configure the maximum number of buffers for storing asynchronous messages for the terminal after which messages are dropped.

When synchronous logging of unsolicited messages and **debug** command output is enabled, unsolicited device output appears on the console or printed after solicited device output appears or is printed. Unsolicited messages and **debug** command output appears on the console after the prompt for user input is returned. Therefore, unsolicited messages and **debug** command output are not interspersed with solicited device output and prompts. After the unsolicited messages appear, the console again displays the user prompt.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**line** [**console** | **vty**] *line-number* [*ending-line-number*] **Example:** `Device(config)# line console`	Specifies the line to be configured for synchronous logging of messages. • **console** —Specifies configurations that occur through the switch console port or the Ethernet management port. • **line vty** *line-number*—Specifies which vty lines are to have synchronous logging enabled. You use a vty connection for configurations that occur through a Telnet session. The range of line numbers is from 0 to 15. You can change the setting of all 16 vty lines at once by entering: line vty 0 15 You can also change the setting of the single vty line being used for your current connection. For example, to change the setting for vty line 2, enter: line vty 2 When you enter this command, the mode changes to line configuration.
Step 3	**logging synchronous** [**level** [*severity-level* | **all**] | **limit** *number-of-buffers*] **Example:** `Device(config)# logging synchronous level 3 limit 1000`	Enables synchronous logging of messages. • (Optional) **level** *severity-level*—Specifies the message severity level. Messages with a severity level equal to or higher than this value are printed asynchronously. Low numbers mean greater severity and high numbers mean lesser severity. The default is 2. • (Optional) **level all**—Specifies that all messages are printed asynchronously regardless of the severity level.

	Command or Action	Purpose
		• (Optional) **limit** *number-of-buffers*—Specifies the number of buffers to be queued for the terminal after which new messages are dropped. The range is 0 to 2147483647. The default is 20.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Disabling Message Logging

Message logging is enabled by default. It must be enabled to send messages to any destination other than the console. When enabled, log messages are sent to a logging process, which logs messages to designated locations asynchronously to the processes that generated the messages.

Disabling the logging process can slow down the switch because a process must wait until the messages are written to the console before continuing. When the logging process is disabled, messages appear on the console as soon as they are produced, often appearing in the middle of command output.

The **logging synchronous** global configuration command also affects the display of messages to the console. When this command is enabled, messages appear only after you press **Return**.

To reenable message logging after it has been disabled, use the **logging on** global configuration command.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**no logging console** **Example:** `Device(config)# no logging console`	Disables message logging.
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Enabling and Disabling Time Stamps on Log Messages

By default, log messages are not time-stamped.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 2	Use one of these commands:
	• **service timestamps log uptime**
	• **service timestamps log datetime**[**msec** | **localtime** | **show-timezone**]
	Example:
	`Device(config)# service timestamps log uptime`
	or
	`Device(config)# service timestamps log datetime`
		Enables log time stamps.
		• **log uptime**—Enables time stamps on log messages, showing the time since the system was rebooted.
		• **log datetime**—Enables time stamps on log messages. Depending on the options selected, the time stamp can include the date, time in milliseconds relative to the local time zone, and the time zone name.

Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Enabling and Disabling Sequence Numbers in Log Messages

If there is more than one log message with the same time stamp, you can display messages with sequence numbers to view these messages. By default, sequence numbers in log messages are not displayed.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 2	**service sequence-numbers** **Example:** `Device(config)# service sequence-numbers`	Enables sequence numbers.
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Defining the Message Severity Level

Limit messages displayed to the selected device by specifying the severity level of the message.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**logging console** *level* **Example:** `Device(config)# logging console 3`	Limits messages logged to the console. By default, the console receives debugging messages and numerically lower levels.
Step 3	**logging monitor** *level* **Example:** `Device(config)# logging monitor 3`	Limits messages logged to the terminal lines. By default, the terminal receives debugging messages and numerically lower levels.
Step 4	**logging trap** *level* **Example:** `Device(config)# logging trap 3`	Limits messages logged to the syslog servers. By default, syslog servers receive informational messages and numerically lower levels.
Step 5	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	

Limiting Syslog Messages Sent to the History Table and to SNMP

This task explains how to limit syslog messages that are sent to the history table and to SNMP.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**logging history** *level* **Example:** `Device(config)# logging history 3`	Changes the default level of syslog messages stored in the history file and sent to the SNMP server. By default, **warnings**, **errors**, **critical**, **alerts**, and **emergencies** messages are sent.
Step 3	**logging history size** *number* **Example:** `Device(config)# logging history size 200`	Specifies the number of syslog messages that can be stored in the history table. The default is to store one message. The range is 0 to 500 messages.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Logging Messages to a UNIX Syslog Daemon

This task is optional.

Note Some recent versions of UNIX syslog daemons no longer accept by default syslog packets from the network. If this is the case with your system, use the UNIX **man syslogd** command to decide what options must be added to or removed from the syslog command line to enable logging of remote syslog messages.

Before you begin

- Log in as root.
- Before you can send system log messages to a UNIX syslog server, you must configure the syslog daemon on a UNIX server.

Procedure

	Command or Action	Purpose
Step 1	Add a line to the file /etc/syslog.conf. **Example:** `local7.debug /usr/adm/logs/cisco.log`	• **local7**—Specifies the logging facility. • **debug**—Specifies the syslog level. The file must already exist, and the syslog daemon must have permission to write to it.

Step 2	Enter these commands at the UNIX shell prompt.
	Example:
	`$ touch /var/log/cisco.log`
	`$ chmod 666 /var/log/cisco.log`
		Creates the log file. The syslog daemon sends messages at this level or at a more severe level to this file.

Step 3	Make sure the syslog daemon reads the new changes. **Example:** `$ kill -HUP `cat /etc/syslog.pid``	For more information, see the **man syslog.conf** and **man syslogd** commands on your UNIX system.

Monitoring and Maintaining System Message Logs

Monitoring Configuration Archive Logs

Command	Purpose
show archive log config {**all** | **number** [*end-number*] | **user** *username* [**session** *number*] *number* [*end-number*] | **statistics**} [**provisioning**]	Displays the entire configuration log or the log for specified parameters.

Configuration Examples for System Message Logs

Example: Switch System Message

This example shows a partial switch system message on a switch:

```
00:00:46: %LINK-3-UPDOWN: Interface Port-channel1, changed state to up
00:00:47: %LINK-3-UPDOWN: Interface GigabitEthernet0/1, changed state to up
00:00:47: %LINK-3-UPDOWN: Interface GigabitEthernet0/2, changed state to up
00:00:48: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed state to down
00:00:48: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state
 to down 2
*Mar  1 18:46:11: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
18:47:02: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Mar  1 18:48:50.483 UTC: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
```

CHAPTER 8

Configuring Online Diagnostics

Information About Configuring Online Diagnostics

With online diagnostics, you can test and verify the hardware functionality of a device while the device is connected to a live network. Online diagnostics contains packet-switching tests that check different hardware components and verify the data path and control signals.

Online diagnostics detects problems in these areas:

- Hardware components
- Interfaces (Ethernet ports and so forth)
- Solder joints

Online diagnostics are categorized as on-demand, scheduled, or health-monitoring diagnostics. On-demand diagnostics run from the CLI; scheduled diagnostics run at user-designated intervals or at specified times when the device is connected to a live network; and health-monitoring runs in the background with user-defined intervals. The health-monitoring test runs every 90, 100, or 150 seconds based on the test.

After you configure online diagnostics, you can manually start diagnostic tests or display the test results. You can also see which tests are configured for the device and the diagnostic tests that have already run.

Generic Online Diagnostics (GOLD) Tests

Note

- Before you enable online diagnostics tests, enable console logging to see all the warning messages.
- While tests are running, all the ports are shut down because a stress test is being performed with looping ports internally, and external traffic might affect the test results. Reboot the switch to bring it to normal operation. When you run the command to reload a switch, the system will ask you if the configuration should be saved. Do not save the configuration.
- If you are running tests on other modules, after a test is initiated and complete, you must reset the module.

The following sections provide information about GOLD tests.

DiagGoldPktTest

This GOLD packet loopback test verifies the MAC-level loopback functionality. In this test, a GOLD packet is sent, for which Unified Access Data Plane (UADP) ASIC provides support in the hardware. The packet loops back at MAC-level and is matched against the stored packet.

Attribute	Description
Disruptive or Nondisruptive	Nondisruptive.
Recommendation	Run this on-demand test as per requirement.
Default	Off.
Corrective action	–
Hardware support	All modules.

DiagThermalTest

This test verifies the temperature reading from a device sensor.

Attribute	Description
Disruptive or Nondisruptive	Nondisruptive.
Recommendation	Do not disable. Run this as an on-demand test, and as a health-monitoring test if the administrator is down.
Default	On.
Corrective action	–
Hardware support	All modules.

DiagPhyLoopbackTest

This PHY loopback test verifies the PHY-level loopback functionality. In this test, a packet, which loops back at the PHY level and is matched against the stored packet, is sent. It cannot be run as a health-monitoring test.

Note In certain cases when this test is run on-demand, ports are moved to the error-disabled state. In such cases, use the **shut** and **no shut** command in interface configuration mode to reenable these ports.

Attribute	Description
Disruptive or Nondisruptive	Disruptive.
Recommendation	If the link to the external connector is down, run this on-demand test to check the health of the link.
Default	Off.
Corrective action	–
Hardware support	All modules.

DiagScratchRegisterTest

This Scratch Register test monitors the health of ASICs by writing values into registers, and reading back the values from these registers.

Attribute	Description
Disruptive or Nondisruptive	Nondisruptive.
Recommendation	Do not disable. Run this test if the task of writing values to the registers fails. This can be run as a health-monitoring test and also as an on-demand test.
Default	On.
Corrective action	–
Hardware support	All modules.

DiagStackCableTest

This test verifies the stack-ring loopback functionality in the stacking environment. It cannot be run as a health-monitoring test.

Attribute	Description
Disruptive or Nondisruptive	Disruptive.
Recommendation	Run this test to verify the stack-ring loopback functionality in the stacking environment.
Default	Off.
Corrective action	If the test fails, check the stack cables and connectors.
Hardware support	All modules.

DiagMemoryTest

This exhaustive ASIC memory test is run during normal device operation. The device utilizes memory built in self-test for this test. The memory test requires device reboot after the test.

Attribute	Description
Disruptive or Nondisruptive	Very disruptive.
Recommendation	Run this on-demand test only if you experience memory-related problems in the system. Do not run this test if you do not want to reload the Supervisor engine.
Default	Off.
Corrective action	–
Hardware support	All modules.

TestUnusedPortLoopback

This test verifies the PHY-level loopback functionality for admin-down ports. In this test, a packet which loops back at the PHY level and is matched against the stored packet, is sent.

Attribute	Description
Disruptive or Nondisruptive	Nondisruptive.
Recommendation	This can be run as a health-monitoring test and also as an on-demand test.
Default	Off.
Corrective action	Displays a syslog message if the test fails for a port.
Hardware support	All modules.

How to Configure Online Diagnostics

The following sections provide information about the various procedures that comprise the online diagnostics configuration.

Starting Online Diagnostic Tests

After you configure diagnostic tests to run on a device, use the **diagnostic start** privileged EXEC command to begin diagnostic testing.

After starting the tests, you cannot stop the testing process midway.

Use the **diagnostic start switch** privileged EXEC command to manually start online diagnostic testing:

Procedure

	Command or Action	Purpose
Step 1	**diagnostic start switch** *number* **test** {*name* | *test-id* | *test-id-range* | **all** | **basic** | **complete** | **minimal** | **non-disruptive** | **per-port**} **Example:** `Device# diagnostic start module 2 test basic`	Starts the diagnostic tests. You can specify the tests by using one of these options: • *name*: Enters the name of the test. • *test-id*: Enters the ID number of the test. • *test-id-range*: Enters the range of test IDs by using integers separated by a comma and a hyphen. • **all**: Starts all of the tests. • **basic**: Starts the basic test suite. • **complete**: Starts the complete test suite. • **minimal**: Starts the minimal bootup test suite. • **non-disruptive**: Starts the nondisruptive test suite. • **per-port**: Starts the per-port test suite.

Configuring Online Diagnostics

You must configure the failure threshold and the interval between tests before enabling diagnostic monitoring.

Scheduling Online Diagnostics

You can schedule online diagnostics to run at a designated time of day, or on a daily, weekly, or monthly basis for a device. Use the **no** form of the **diagnostic schedule switch** command to remove the scheduling.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device #configure terminal`	Enters global configuration mode.
Step 2	**diagnostic schedule** *number* **test** {*name* | *test-id* | *test-id-range* | **all** | **basic** | **complete** | **minimal** | **non-disruptive** | **per-port**} {**daily** | **on** *mm dd yyyy hh:mm* | **port** *inter-port-number port-number-list* | **weekly** *day-of-week hh:mm*} **Example:** `Device(config)# diagnostic schedule 3 test 1-5 on July 3 2013 23:10`	Schedules on-demand diagnostic test for a specific day and time. When specifying the test to be scheduled, use these options: • *name*: Name of the test that appears in the **show diagnostic content** command output. • *test-id*: ID number of the test that appears in the **show diagnostic content** command output.

	Command or Action	Purpose
		• *test-id-range*: ID numbers of the tests that appear in the **show diagnostic content** command output. • **all**: All test IDs. • **basic**: Starts the basic on-demand diagnostic tests. • **complete**: Starts the complete test suite. • **minimal**: Starts the minimal bootup test suite. • **non-disruptive**: Starts the nondisruptive test suite. • **per-port**: Starts the per-port test suite. You can schedule the tests as follows: • Daily: Use the **daily** *hh:mm* parameter. • Specific day and time: Use the **on** *mm dd yyyy hh:mm* parameter. • Weekly: Use the **weekly** *day-of-week hh:mm* parameter.

Configuring Health-Monitoring Diagnostics

You can configure health-monitoring diagnostic testing on a device while it is connected to a live network. You can configure the execution interval for each health-monitoring test, enable the device to generate a syslog message because of a test failure, and enable a specific test.

Use the **no** form of this command to disable testing.

By default, health monitoring is enabled only for a few tests, and the device generates a syslog message when a test fails.

Follow these steps to configure and enable the health-monitoring diagnostic tests:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**diagnostic monitor interval switch** *number* **test** {*name* | *test-id* | *test-id-range* | **all**} *hh:mm:ss milliseconds day* **Example:** `Device(config)# diagnostic monitor interval switch 2 test 1 12:30:00 750 5`	Configures the health-monitoring interval of the specified test. When specifying a test, use one of these parameters: • *name*: Name of the test that appears in the **show diagnostic content** command output. • *test-id*: ID number of the test that appears in the **show diagnostic content** command output. • *test-id-range*: ID numbers of the tests that appear in the **show diagnostic content** command output. • **all**: All the diagnostic tests. When specifying the interval, set these parameters: • *hh:mm:ss*: Monitoring interval, in hours, minutes, and seconds. The range for *hh* is 0 to 24, and the range for *mm* and *ss* is 0 to 60. • *milliseconds*: Monitoring interval, in milliseconds (ms). The range is from 0 to 999. • *day*: Monitoring interval, in number of days. The range is from 0 to 20.
Step 4	**diagnostic monitor syslog** **Example:** `Device(config)# diagnostic monitor syslog`	(Optional) Configures the switch to generate a syslog message when a health-monitoring test fails.
Step 5	**diagnostic monitor threshold switch** *number number* **test** {*name* | *test-id* | *test-id-range* | **all**} **failure count** *count* **Example:** `Device(config)# diagnostic monitor threshold switch 2 test 1 failure count 20`	(Optional) Sets the failure threshold for the health-monitoring test. When specifying the tests, use one of these parameters: • *name*: Name of the test that appears in the **show diagnostic content** command output. • *test-id*: ID number of the test that appears in the **show diagnostic content** command output. • *test-id-range*: ID numbers of the tests that appear in the **show diagnostic content** command output. • **all**: All the diagnostic tests. The range for the failure threshold *count* is 0 to 99.
Step 6	**diagnostic monitor switch***number* **test** {*name* | *test-id* | *test-id-range* | **all**} **Example:**	Enables the specified health-monitoring tests. The **switch** *number* keyword is supported only on stacking switches, and does not apply to the ESS9300.

	Command or Action	Purpose
	`Device(config)# diagnostic monitor switch 2 test 1`	When specifying the tests, use one of these parameters: • *name*: Name of the test that appears in the **show diagnostic content** command output. • *test-id*: ID number of the test that appears in the **show diagnostic content** command output. • *test-id-range*: ID numbers of the tests that appear in the **show diagnostic content** command output. • **all**: All the diagnostic tests.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show diagnostic** { **content** | **post** | **result** | **schedule** | **status** | **switch** }	(Optional) Display the online diagnostic test results and the supported test suites.
Step 9	**show running-config** **Example:** `Device# show running-config`	(Optional) Verifies your entries.
Step 10	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring and Maintaining Online Diagnostics

You can display the online diagnostic tests that are configured for a device and check the test results by using the privileged EXEC **show** commands in this table:

Table 13: Commands for Diagnostic Test Configuration and Results

Command	Purpose
show diagnostic content switch [*number* | **all**]	Displays the online diagnostics configured for a switch.
show diagnostic status	Displays the diagnostic tests that are running currently. .

Command	Purpose
show diagnostic result switch [*number* | **all**] [**detail** | **test** {*name* | *test-id* | *test-id-range* | **all**} [**detail**]]	Displays the online diagnostics test results.
show diagnostic switch [*number* | **all**] [**detail**]	Displays the online diagnostics test results.
show diagnostic schedule [*number* | **all**]	Displays the online diagnostics test schedule.
show diagnostic post	Displays the POST results. (The output is the same as the **show post** command output.)
show diagnostic events {**event-type** | **module**}	Displays diagnostic events such as error, information, or warning based on the test result.
show diagnostic description module [**number**] **test** { **name** | **test-id** | **all** }	Displays the short description of the results from an individual test or all the tests.

Configuration Examples for Online Diagnostics

The following sections provide examples of online diagnostics configurations.

Examples: Start Diagnostic Tests

This example shows how to start a diagnostic test by using the test name:

```
Device# diagnostic start switch 1 test DiagPOETest
```

This example shows how to start all of the basic diagnostic tests:

```
Device# diagnostic start switch 1 test all
```

Example: Configure a Health-Monitoring Test

This example shows how to configure a health-monitoring test:

```
Device(config)#
Device(config)#
```

Example: Schedule Diagnostic Test

This example shows how to schedule diagnostic testing for a specific day and time on a specific switch:

```
Device(config)#
```

This example shows how to schedule diagnostic testing to occur weekly at a certain time on a specific switch:

```
Device(config)#
```

Example: Displaying Online Diagnostics

This example shows how to display on-demand diagnostic settings:

```
Device# show diagnostic ondemand settings

Test iterations = 1
Action on test failure = continue
```

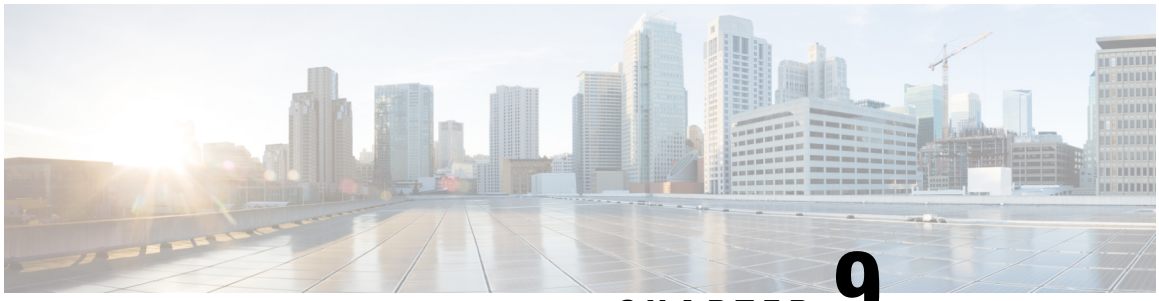
This example shows how to display diagnostic events for errors:

```
Device# show diagnostic events event-type error

Diagnostic events (storage for 500 events, 0 events recorded)
Number of events matching above criteria = 0

No diagnostic log entry exists.
```

This example shows how to display the description for a diagnostic test:

```
Device# show diagnostic description switch 1 test all

DiagGoldPktTest :
        The GOLD packet Loopback test verifies the MAC level loopback
        functionality. In this test, a GOLD packet, for which doppler
        provides the support in hardware, is sent. The packet loops back
        at MAC level and is matched against the stored packet. It is a non
        -disruptive test.

DiagThermalTest :
        This test verifies the temperature reading from the sensor is below the yellow
        temperature threshold. It is a non-disruptive test and can be run as a health
monitoring test.

DiagFanTest :
        This test verifies all fan modules have been inserted and working properly on the
board
        It is a non-disruptive test and can be run as a health monitoring test.

DiagPhyLoopbackTest :
        The PHY Loopback test verifies the PHY level loopback
        functionality. In this test, a packet is sent which loops back
        at PHY level and is matched against the stored packet. It is a
        disruptive test and cannot be run as a health monitoring test.

DiagScratchRegisterTest :
        The Scratch Register test monitors the health of application-specific
        integrated circuits (ASICs) by writing values into registers and reading
        back the values from these registers. It is a non-disruptive test and can
        be run as a health monitoring test.

DiagPoETest :
        This test checks the PoE controller functionality. This is a disruptive test
        and should not be performed during normal switch operation.


DiagMemoryTest :
        This test runs the exhaustive ASIC memory test during normal switch operation
        NG3K utilizes mbist for this test. Memory test is very disruptive
        in nature and requires switch reboot after the test.
```

```
Device#
```



CHAPTER 9

Managing Configuration Files

Prerequisites for Managing Configuration Files

- You should have at least a basic familiarity with the Cisco IOS environment and the command-line interface.
- You should have at least a minimal configuration running on your system. You can create a basic configuration file using the **setup** command.

Restrictions for Managing Configuration Files

- Many of the Cisco IOS commands described in this document are available and function only in certain configuration modes on the device.
- Some of the Cisco IOS configuration commands are only available on certain device platforms, and the command syntax may vary on different platforms.

Information About Managing Configuration Files

Types of Configuration Files

Configuration files contain the Cisco IOS software commands used to customize the functionality of your Cisco device. Commands are parsed (translated and executed) by the Cisco IOS software when the system is booted (from the startup-config file) or when you enter commands at the CLI in a configuration mode.

Startup configuration files (startup-config) are used during system startup to configure the software. Running configuration files (running-config) contain the current configuration of the software. The two configuration files can be different. For example, you may want to change the configuration for a short time period rather

than permanently. In this case, you would change the running configuration using the **configure terminal** EXEC command but not save the configuration using the **copy running-config startup-config** EXEC command.

To change the running configuration, use the **configure terminal** command, as described in the Modifying the Configuration File, on page 133 section. As you use the Cisco IOS configuration modes, commands generally are executed immediately and are saved to the running configuration file either immediately after you enter them or when you exit a configuration mode.

To change the startup configuration file, you can either save the running configuration file to the startup configuration using the **copy running-config startup-config** EXEC command or copy a configuration file from a file server to the startup configuration (see the Copying a Configuration File from a TFTP Server to the Device section for more information).

Configuration Mode and Selecting a Configuration Source

To enter configuration mode on the device, enter the **configure** command at the privileged EXEC prompt. The Cisco IOS software responds with the following prompt asking you to specify the terminal, memory, or a file stored on a network server (network) as the source of configuration commands:

```
Configuring from terminal, memory, or network [terminal]?
```

Configuring from the terminal allows you to enter configuration commands at the command line, as described in the following section. See the Re-executing the Configuration Commands in the Startup Configuration File section for more information.

Configuring from the network allows you to load and execute configuration commands over the network. See the Copying a Configuration File from a TFTP Server to the Device section for more information.

Configuration File Changes Using the CLI

The Cisco IOS software accepts one configuration command per line. You can enter as many configuration commands as you want. You can add comments to a configuration file describing the commands you have entered. Precede a comment with an exclamation point (!). Because comments are *not* stored in NVRAM or in the active copy of the configuration file, comments do not appear when you list the active configuration with the **show running-config** or **more system:running-config** EXEC command. Comments are not displayed when you list the startup configuration with the **show startup-config** or **more nvram:startup-config** EXEC mode command. Comments are stripped out of the configuration file when it is loaded onto the device. However, you can list the comments in configuration files stored on a File Transfer Protocol (FTP), Remote Copy Protocol (RCP), or Trivial File Transfer Protocol (TFTP) server. When you configure the software using the CLI, the software executes the commands as you enter them.

Location of Configuration Files

Configuration files are stored in the following locations:

- The running configuration is stored in RAM.
- On all platforms except the Class A Flash file system platforms, the startup configuration is stored in nonvolatile random-access memory (NVRAM).

- On Class A Flash file system platforms, the startup configuration is stored in the location specified by the CONFIG_FILE environment variable (see the Specifying the CONFIG_FILE Environment Variable on Class A Flash File Systems , on page 153 section). The CONFIG_FILE variable defaults to NVRAM and can be a file in the following file systems:
  - **nvram:** (NVRAM)
  - **flash:** (internal flash memory)
  - **usbflash0:** (external usbflash file system)

Copy Configuration Files from a Network Server to the Device

You can copy configuration files from a TFTP, rcp, or FTP server to the running configuration or startup configuration of the device. You may want to perform this function for one of the following reasons:

- To restore a backed-up configuration file.
- To use the configuration file for another device. For example, you may add another device to your network and want it to have a similar configuration to the original device. By copying the file to the new device, you can change the relevant parts rather than recreating the whole file.
- To load the same configuration commands on to all of the devices in your network so that all of the devices have similar configurations.

The **copy**{**ftp:** | **rcp:** | **tftp:system:running-config**} EXEC command loads the configuration files into the device as if you were typing the commands on the command line. The device does not erase the existing running configuration before adding the commands. If a command in the copied configuration file replaces a command in the existing configuration file, the existing command is erased. For example, if the copied configuration file contains a different IP address in a particular command than the existing configuration, the IP address in the copied configuration is used. However, some commands in the existing configuration may not be replaced or negated. In this case, the resulting configuration file is a mixture of the existing configuration file and the copied configuration file, with the copied configuration file having precedence.

To restore a configuration file to an exact copy of a file stored on a server, you need to copy the configuration file directly to the startup configuration (using the **copy ftp:**| **rcp:**| **tftp:**} **nvram:startup-config** command) and reload the device.

To copy configuration files from a server to a device, perform the tasks described in the following sections.

The protocol that you use depends on which type of server you are using. The FTP and rcp transport mechanisms provide faster performance and more reliable delivery of data than TFTP. These improvements are possible because the FTP and rcp transport mechanisms are built on and use the TCP/IP stack, which is connection-oriented.

Copying a Configuration File from the Device to a TFTP Server

In some implementations of TFTP, you must create a dummy file on the TFTP server and give it read, write, and execute permissions before copying a file over it. Refer to your TFTP documentation for more information.

Copying a Configuration File from the Device to an RCP Server

You can copy a configuration file from the device to an RCP server.

One of the first attempts to use the network as a resource in the UNIX community resulted in the design and implementation of the remote shell protocol, which included the remote shell (rsh) and remote copy (rcp) functions. Rsh and rcp give users the ability to execute commands remotely and copy files to and from a file system residing on a remote host or server on the network. The Cisco implementation of rsh and rcp interoperates with standard implementations.

The rcp **copy** commands rely on the rsh server (or daemon) on the remote system. To copy files using rcp, you need not create a server for file distribution, as you do with TFTP. You need only to have access to a server that supports the remote shell (rsh). (Most UNIX systems support rsh.) Because you are copying a file from one place to another, you must have read permission on the source file and write permission on the destination file. If the destination file does not exist, rcp creates it for you.

Although the Cisco rcp implementation emulates the functions of the UNIX rcp implementation—copying files among systems on the network—the Cisco command syntax differs from the UNIX rcp command syntax. The Cisco rcp support offers a set of **copy** commands that use rcp as the transport mechanism. These rcp **copy** commands are similar in style to the Cisco TFTP **copy** commands, but they offer an alternative that provides faster performance and reliable delivery of data. These improvements are possible because the rcp transport mechanism is built on and uses the TCP/IP stack, which is connection-oriented. You can use rcp commands to copy system images and configuration files from the device to a network server and vice versa.

You also can enable rcp support to allow users on remote systems to copy files to and from the device.

To configure the Cisco IOS software to allow remote users to copy files to and from the device, use the **ip rcmd rcp-enable** global configuration command.

Restrictions

The RCP protocol requires a client to send a remote username on each RCP request to a server. When you copy a configuration file from the device to a server using RCP, the Cisco IOS software sends the first valid username it encounters in the following sequence:

1. The username specified in the **copy** EXEC command, if a username is specified.
2. The username set by the **ip rcmd remote-username** global configuration command, if the command is configured.
3. The remote username associated with the current tty (terminal) process. For example, if the user is connected to the device through Telnet and was authenticated through the **username** command, the device software sends the Telnet username as the remote username.
4. The device host name.

For the RCP copy request to execute successfully, an account must be defined on the network server for the remote username. If the server has a directory structure, the configuration file or image is written to or copied from the directory associated with the remote username on the server. For example, if the system image resides in the home directory of a user on the server, you can specify that user name as the remote username.

Use the **ip rcmd remote-username** command to specify a username for all copies. (Rcmd is a UNIX routine used at the super-user level to execute commands on a remote machine using an authentication scheme based on reserved port numbers. Rcmd stands for “remote command”). Include the username in the **copy** command if you want to specify a username for that copy operation only.

If you are writing to the server, the RCP server must be properly configured to accept the RCP write request from the user on the device. For UNIX systems, you must add an entry to the .rhosts file for the remote user on the RCP server. For example, suppose the device contains the following configuration lines:

```
hostname Device1
ip rcmd remote-username User0
```

If the device IP address translates to device1.example.com, then the .rhosts file for User0 on the RCP server should contain the following line:

```
Device1.example.com Device1
```

Requirements for the RCP Username

The RCP protocol requires a client to send a remote username on each RCP request to a server. When you copy a configuration file from the device to a server using RCP, the Cisco IOS software sends the first valid username it encounters in the following sequence:

1. The username specified in the **copy** EXEC command, if a username is specified.
2. The username set by the **ip rcmd remote-username** global configuration command, if the command is configured.
3. The remote username associated with the current tty (terminal) process. For example, if the user is connected to the device through Telnet and is authenticated through the **username** command, the device software sends the Telnet username as the remote username.
4. The device host name.

For the RCP copy request to execute, an account must be defined on the network server for the remote username. If the server has a directory structure, the configuration file or image is written to or copied from the directory associated with the remote username on the server. For example, if the system image resides in the home directory of a user on the server, specify that user name as the remote username.

Refer to the documentation for your RCP server for more information.

Copying a Configuration File from the Device to an FTP Server

You can copy a configuration file from the device to an FTP server.

Understanding the FTP Username and Password

Note The password must not contain the special character '@'. If the character '@' is used, the copy fails to parse the IP address of the server.

The FTP protocol requires a client to send a remote username and password on each FTP request to a server. When you copy a configuration file from the device to a server using FTP, the Cisco IOS software sends the first valid username it encounters in the following sequence:

1. The username specified in the **copy** EXEC command, if a username is specified.
2. The username set by the **ip ftp username** global configuration command, if the command is configured.
3. Anonymous.

The device sends the first valid password it encounters in the following sequence:

1. The password specified in the **copy** command, if a password is specified.
2. The password set by the **ip ftp password** command, if the command is configured.
3. The device forms a password *username @devicename.domain* . The variable *username* is the username associated with the current session, *devicename* is the configured host name, and *domain* is the domain of the device.

The username and password must be associated with an account on the FTP server. If you are writing to the server, the FTP server must be properly configured to accept the FTP write request from the user on the device.

If the server has a directory structure, the configuration file or image is written to or copied from the directory associated with the username on the server. For example, if the system image resides in the home directory of a user on the server, specify that user name as the remote username.

Refer to the documentation for your FTP server for more information.

Use the **ip ftp username** and **ip ftp password** global configuration commands to specify a username and password for all copies. Include the username in the **copy** EXEC command if you want to specify a username for that copy operation only.

Copying files through a VRF

You can copy files through a VRF interface specified in the **copy** command. Specifying the VRF in the **copy** command is easier and more efficient as you can directly change the source interface without using a change request for the configuration.

Example

The following example shows how to copy files through a VRF, using the **copy** command:

```
Device# copy scp: bootflash-1: vrf test-vrf
Address or name of remote host [10.1.2.3]?
Source username [ScpUser]?
Source filename [/auto/tftp-server/ScpUser/vrf_test.txt]?
Destination filename [vrf_test.txt]?
Getting the vrf name as test-vrf
Password:
Sending file modes: C0644 10 vrf_test.txt
!
223 bytes copied in 22.740 secs (10 bytes/sec)
```

Copy Configuration Files from a Switch to Another Switch

You can copy the configurations from one switch to another. This is a 2-step process - Copy the configurations from the switch to the TFTP server, and then from TFTP to another switch.

To copy your current configurations from the switch, run the command **copy startup-config tftp:** and follow the instructions. The configurations are copied onto the TFTP server.

Then, login to another switch and run the command **copy tftp: startup-config** and follow the instructions. The configurations are now copied onto the other switch.

After the configurations are copied, to save your configurations, use **write memory** command and then either reload the switch or run the **copy startup-config running-config** command

Configuration Files Larger than NVRAM

To maintain a configuration file that exceeds the size of NVRAM, you should be aware of the information in the following sections.

Compressing the Configuration File

The **service compress-config** global configuration command specifies that the configuration file be stored compressed in NVRAM. Once the configuration file has been compressed, the device functions normally. When the system is booted, it recognizes that the configuration file is compressed, expands it, and proceeds normally. The **more nvram:startup-config** EXEC command expands the configuration before displaying it.

Before you compress configuration files, refer to the appropriate hardware installation and maintenance publication. Verify that your system's ROMs support file compression. If not, you can install new ROMs that support file compression.

The size of the configuration must not exceed three times the NVRAM size. For a 128-KB size NVRAM, the largest expanded configuration file size is 384 KB.

The **service compress-config** global configuration command works only if you have Cisco IOS software Release 10.0 or later release boot ROMs. Installing new ROMs is a one-time operation and is necessary only if you do not already have Cisco IOS Release 10.0 in ROM. If the boot ROMs do not recognize a compressed configuration, the following message is displayed:

```
Boot ROMs do not support NVRAM compression Config NOT written to NVRAM
```

Storing the Configuration in Flash Memory on Class A Flash File Systems

On class A Flash file system devices, you can store the startup configuration in flash memory by setting the **CONFIG_FILE** environment variable to a file in internal flash memory or flash memory in a PCMCIA slot.

See the Specifying the CONFIG_FILE Environment Variable on Class A Flash File Systems , on page 153 section for more information.

Care must be taken when editing or changing a large configuration. Flash memory space is used every time a **copy system:running-config nvram:startup-config** EXEC command is issued. Because file management for flash memory (such as optimizing free space) is not done automatically, you must pay close attention to available flash memory. Use the **squeeze** command to reclaim used space. We recommend that you use a large-capacity Flash card of at least 20 MB.

Loading the Configuration Commands from the Network

You can also store large configurations on FTP, RCP, or TFTP servers and download them at system startup. To use a network server to store large configurations, see the Copying a Configuration File from the Device to a TFTP Server , on page 134 and Configuring the Device to Download Configuration Files, on page 131 sections for more information on these commands.

Configuring the Device to Download Configuration Files

You can configure the device to load one or two configuration files at system startup. The configuration files are loaded into memory and read in as if you were typing the commands at the command line. Thus, the configuration for the device is a mixture of the original startup configuration and the one or two downloaded configuration files.

Network Versus Host Configuration Files

For historical reasons, the first file the device downloads is called the network configuration file. The second file the device downloads is called the host configuration file. Two configuration files can be used when all of the devices on a network use many of the same commands. The network configuration file contains the standard commands used to configure all of the devices. The host configuration files contain the commands specific to one particular host. If you are loading two configuration files, the host configuration file should be the configuration file you want to have precedence over the other file. Both the network and host configuration files must reside on a network server reachable via TFTP, RCP, or FTP, and must be readable.

How to Manage Configuration File Information

Displaying Configuration File Information

To display information about configuration files, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show boot** **Example:** `Device# show boot`	Lists the contents of the BOOT environment variable (if set), the name of the configuration file pointed to by the CONFIG_FILE environment variable, and the contents of the BOOTLDR environment variable.
Step 3	**more** *file-url* **Example:** `Device# more 10.1.1.1`	Displays the contents of a specified file.
Step 4	**show running-config** **Example:** `Device# show running-config`	Displays the contents of the running configuration file. (Command alias for the **more system:running-config** command.)
Step 5	**show startup-config** **Example:** `Device# show startup-config`	Displays the contents of the startup configuration file. (Command alias for the **more nvram:startup-config** command.) The CONFIG_FILE variable defaults to NVRAM.

Modifying the Configuration File

The Cisco IOS software accepts one configuration command per line. You can enter as many configuration commands as you want. You can add comments to a configuration file describing the commands you have entered. Precede a comment with an exclamation point (!). Because comments are *not* stored in NVRAM or in the active copy of the configuration file, comments do not appear when you list the active configuration with the **show running-config** or **more system:running-config** EXEC commands. Comments do not display when you list the startup configuration with the **show startup-config** or **more nvram:startup-config** EXEC mode commands. Comments are stripped out of the configuration file when it is loaded onto the device. However, you can list the comments in configuration files stored on a File Transfer Protocol (FTP), Remote Copy Protocol (RCP), or Trivial File Transfer Protocol (TFTP) server. When you configure the software using the CLI, the software executes the commands as you enter them. To configure the software using the CLI, use the following commands in privileged EXEC mode:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**configuration command** **Example:** `Device(config)# configuration command`	Enter the necessary configuration commands. The Cisco IOS documentation set describes configuration commands organized by technology.
Step 4	Do one of the following: • **end** • **^Z** **Example:** `Device(config)# end`	Ends the configuration session and exits to EXEC mode. **Note** When you press the Ctrl and Z keys simultaneously, ^Z is displayed to the screen.
Step 5	**copy system:running-config nvram:startup-config** **Example:** `Device# copy system:running-config nvram:startup-config`	Saves the running configuration file as the startup configuration file. You may also use the **copy running-config startup-config** command alias, but you should be aware that this command is less precise. On most platforms, this command saves the configuration to NVRAM. On the Class A Flash file system platforms, this step saves the configuration to the location specified by the CONFIG_FILE environment variable (the

	Command or Action	Purpose
		default CONFIG_FILE variable specifies that the file should be saved to NVRAM).

Examples

In the following example, the device prompt name of the device is configured. The comment line, indicated by the exclamation mark (!), does not execute any command. The **hostname** command is used to change the device name from device to new_name. By pressing Ctrl-Z (^Z) or entering the **end** command, the user quits configuration mode. The **copy system:running-config nvram:startup-config** command saves the current configuration to the startup configuration.

```
Device# configure terminal
Device(config)# !The following command provides the switch host name.
Device(config)# hostname new_name
new_name(config)# end
new_name# copy system:running-config nvram:startup-config
```

When the startup configuration is NVRAM, it stores the current configuration information in text format as configuration commands, recording only non-default settings. The memory is checksummed to guard against corrupted data.

Note Some specific commands might not get saved to NVRAM. You need to enter these commands again if you reboot the machine. These commands are noted in the documentation. We recommend that you keep a list of these settings so that you can quickly reconfigure your device after rebooting.

Copying a Configuration File from the Device to a TFTP Server

To copy configuration information on a TFTP network server, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy system:running-config tftp:** [[[*//location*]*/directory*]*/filename*] **Example:** `Device# copy system:running-config tftp: //server1/topdir/file10`	Copies the running configuration file to a TFTP server.

	Command or Action	Purpose
Step 3	**copy nvram:startup-config tftp:** [[[*//location*]*/directory*]*/filename*] **Example:** `Device# copy nvram:startup-config tftp: //server1/1stdir/file10`	Copies the startup configuration file to a TFTP server.

Examples

The following example copies a configuration file from a device to a TFTP server:

```
Device# copy system:running-config tftp://192.168.2.155/backup-confg
Write file backup-confg on host 192.168.2.155? [confirm] Y
Writing backup-confg!!! [OK]
```

What to Do Next

After you have issued the **copy** command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Copying a Configuration File from the Device to an RCP Server

To copy a startup configuration file or a running configuration file from the device to an RCP server, use the following commands beginning in privileged EXEC mode:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip rcmd remote-username** *username* **Example:** `Device(config)# ip rcmd remote-username NetAdmin1`	(Optional) Changes the default remote username.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode.
Step 5	Do one of the following: • **copy system:running-config rcp:** [[[//[*username@*]*location*]/*directory*]/*filename*] • **copy nvram:startup-config rcp:** [[[//[*username@*]*location*]/*directory*]/*filename*] **Example:** `Device# copy system:running-config rcp: //NetAdmin1@example.com/dir-files/file1`	• Specifies that the device running configuration file is to be stored on an RCP server or • Specifies that the device startup configuration file is to be stored on an RCP server

Examples

Storing a Running Configuration File on an RCP Server

The following example copies the running configuration file named runfile2-confg to the netadmin1 directory on the remote host with an IP address of 192.168.101.101:

```
Device# copy system:running-config rcp://netadmin1@172.16.101.101/runfile2-confg
Write file runfile2-confg on host 192.168.101.101?[confirm]
Building configuration...[OK]
Connected to 192.168.101.101
Device#
```

Storing a Startup Configuration File on an RCP Server

The following example shows how to store a startup configuration file on a server by using RCP to copy the file:

```
Device# configure terminal
 Device(config)# ip rcmd remote-username netadmin2
 Device(config)# end
 Device# copy nvram:startup-config rcp:
 Remote host[]? 192.168.101.101
 Name of configuration file to write [start-confg]?
Write file start-confg on host 192.168.101.101?[confirm]
![OK]
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Copying a Configuration File from the Device to the FTP Server

To copy a startup configuration file or a running configuration file from the device to an FTP server, complete the following tasks:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode on the device.
Step 3	**ip ftp username** *username* **Example:** `Device(config)# ip ftp username NetAdmin1`	(Optional) Specifies the default remote username.
Step 4	**ip ftp password** *password* **Example:** `Device(config)# ip ftp password adminpassword`	(Optional) Specifies the default password.
Step 5	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode. This step is required only if you override the default remote username or password (see Steps 2 and 3).
Step 6	Do one of the following: • **copy system:running-config ftp:** [[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*] or • **copy nvram:startup-config ftp:** [[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*] **Example:** `Device# copy system:running-config ftp:`	Copies the running configuration or startup configuration file to the specified location on the FTP server.

Examples

Storing a Running Configuration File on an FTP Server

The following example copies the running configuration file named runfile-confg to the netadmin1 directory on the remote host with an IP address of 192.168.101.101:

```
Device# copy system:running-config ftp://netadmin1:mypass@192.168.101.101/runfile-confg
Write file runfile-confg on host 192.168.101.101?[confirm]
Building configuration...[OK]
Connected to 192.168.101.101
Device#
```

Storing a Startup Configuration File on an FTP Server

The following example shows how to store a startup configuration file on a server by using FTP to copy the file:

```
Device# configure terminal
 Device(config)# ip ftp username netadmin2
 Device(config)# ip ftp password mypass
 Device(config)# end
 Device# copy nvram:startup-config ftp:
 Remote host[]? 192.168.101.101
 Name of configuration file to write [start-confg]?
Write file start-confg on host 192.168.101.101?[confirm]
![OK]
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Copying a Configuration File from a TFTP Server to the Device

To copy a configuration file from a TFTP server to the device, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy tftp:** [[[*//location*]*/directory*]*/filename*] **system:running-config** **Example:** `Device# copy tftp://server1/dir10/datasource system:running-config`	Copies a configuration file from a TFTP server to the running configuration.
Step 3	**copy tftp:** [[[*//location*]*/directory*]*/filename*] **nvram:startup-config** **Example:**	Copies a configuration file from a TFTP server to the startup configuration.

	Command or Action	Purpose
	`Device# copy tftp://server1/dir10/datasource nvram:startup-config`	
Step 4	**copy tftp:** [[[*//location*]*/directory*]*/filename*]**flash-[n]:/directory/startup-config** **Example:** `Device# copy tftp://server1/dir10/datasource flash:startup-config`	Copies a configuration file from a TFTP server to the startup configuration.

Examples

In the following example, the software is configured from the file named **tokyo-confg** at IP address 192.168.2.155:

```
Device# copy tftp://192.168.2.155/tokyo-confg system:running-config

Configure using tokyo-confg from 192.168.2.155? [confirm] Y

Booting tokyo-confg from 192.168.2.155:!!! [OK - 874/16000 bytes]
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Copying a Configuration File from the rcp Server to the Device

To copy a configuration file from an rcp server to the running configuration or startup configuration, complete the following tasks:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	(Optional) Enters configuration mode from the terminal. This step is required only if you override the default remote username (see Step 3).

	Command or Action	Purpose
Step 3	**ip rcmd remote-username** *username* **Example:** `Device(config)# ip rcmd remote-username NetAdmin1`	(Optional) Specifies the remote username.
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode. This step is required only if you override the default remote username (see Step 2).
Step 5	Do one of the following: • **copy rcp:**[[[//[*username*@]*location*]/*directory*]/*filename*]**system:running-config** • **copy rcp:**[[[//[*username*@]*location*]/*directory*]/*filename*]**nvram:startup-config** **Example:** `Device# copy rcp://[user1@example.com/dir10/fileone] nvram:startup-config`	Copies the configuration file from an rcp server to the running configuration or startup configuration.

Examples

Copy RCP Running-Config

The following example copies a configuration file named host1-confg from the netadmin1 directory on the remote server with an IP address of 192.168.101.101, and loads and runs the commands on the device:

```
device# copy rcp://netadmin1@192.168.101.101/host1-confg system:running-config
Configure using host1-confg from 192.168.101.101? [confirm]
Connected to 192.168.101.101
Loading 1112 byte file host1-confg:![OK]
device#
%SYS-5-CONFIG: Configured from host1-config by rcp from 192.168.101.101
```

Copy RCP Startup-Config

The following example specifies a remote username of netadmin1. Then it copies the configuration file named host2-confg from the netadmin1 directory on the remote server with an IP address of 192.168.101.101 to the startup configuration.

```
device# configure terminal
device(config)# ip rcmd remote-username netadmin1
device(config)# end
device# copy rcp: nvram:startup-config
Address of remote host [255.255.255.255]? 192.168.101.101
Name of configuration file[rtr2-confg]? host2-confg
Configure using host2-confg from 192.168.101.101?[confirm]
Connected to 192.168.101.101
Loading 1112 byte file host2-confg:![OK]
[OK]
```

```
device#
%SYS-5-CONFIG_NV:Non-volatile store configured from host2-config by rcp from 192.168.101.101
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Copying a Configuration File from an FTP Server to the Device

To copy a configuration file from an FTP server to the running configuration or startup configuration, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	(Optional) Allows you to enter global configuration mode. This step is required only if you want to override the default remote username or password (see Steps 3 and 4).
Step 3	**ip ftp username** *username* **Example:** `Device(config)# ip ftp username NetAdmin1`	(Optional) Specifies the default remote username.
Step 4	**ip ftp password** *password* **Example:** `Device(config)# ip ftp password adminpassword`	(Optional) Specifies the default password.
Step 5	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode. This step is required only if you override the default remote username or password (see Steps 3 and 4).
Step 6	Do one of the following: • **copy ftp:** [[[//[*username*[**:***password*]**@**]*location*] /*directory*]/*filename*]**system:running-config** • **copy ftp:** [[[//[*username*[**:***password*]**@**]*location*]/*directory*]/*filename*]**nvram:startup-config** **Example:**	Using FTP copies the configuration file from a network server to running memory or the startup configuration.

	Command or Action	Purpose
	`Device# copy ftp:nvram:startup-config`	

Examples

Copy FTP Running-Config

The following example copies a host configuration file named host1-confg from the netadmin1 directory on the remote server with an IP address of 192.168.101.101, and loads and runs the commands on the device:

```
device# copy ftp://netadmin1:mypass@192.168.101.101/host1-confg system:running-config
Configure using host1-confg from 192.168.101.101? [confirm]
Connected to 192.168.101.101
Loading 1112 byte file host1-confg:![OK]
device#
%SYS-5-CONFIG: Configured from host1-config by ftp from 192.168.101.101
```

Copy FTP Startup-Config

The following example specifies a remote username of netadmin1. Then it copies the configuration file named host2-confg from the netadmin1 directory on the remote server with an IP address of 192.168.101.101 to the startup configuration:

```
device# configure terminal
device(config)# ip ftp username netadmin1
device(config)# ip ftp password mypass
device(config)# end
device# copy ftp: nvram:startup-config
Address of remote host [255.255.255.255]? 192.168.101.101
Name of configuration file[host1-confg]? host2-confg
Configure using host2-confg from 192.168.101.101?[confirm]
Connected to 192.168.101.101
Loading 1112 byte file host2-confg:![OK]
[OK]
device#
%SYS-5-CONFIG_NV:Non-volatile store configured from host2-config by ftp from 192.168.101.101
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Maintaining Configuration Files Larger than NVRAM

To maintain a configuration file that exceeds the size of NVRAM, perform the tasks described in the following sections:

Compressing the Configuration File

To compress configuration files, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service compress-config** **Example:** `Device(config)# service compress-config`	Specifies that the configuration file be compressed.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 5	Do one of the following: • Use FTP, RCP, or TFTP to copy the new configuration. • **configure terminal** **Example:** `Device# configure terminal`	Enters the new configuration: • If you try to load a configuration that is more than three times larger than the NVRAM size, the following error message is displayed: "[buffer overflow - *file-size* /*buffer-size* bytes]."
Step 6	**copy system:running-config nvram:startup-config** **Example:** `Device(config)# copy system:running-config nvram:startup-config`	When you have finished changing the running-configuration, save the new configuration.

Examples

The following example compresses a 129-KB configuration file to 11 KB:

```
Device# configure terminal
Device(config)# service compress-config
Device(config)# end
Device# copy tftp://192.168.2.15/test-confg system:running-config
```

```
Configure using test-confg from 192.168.2.155? [confirm] y

Booting test-confg from 192.168.2.155:!!! [OK - 874/16000 bytes]
Device# copy system:running-config nvram:startup-config

Building configuration...
Compressing configuration from 129648 bytes to 11077 bytes
[OK]
```

Storing the Configuration in Flash Memory on Class A Flash File Systems

To store the startup configuration in flash memory, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy nvram:startup-config** *flash-filesystem:filename* **Example:** `Device# copy nvram:startup-config usbflash0:switch-config`	Copies the current startup configuration to the new location to create the configuration file.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**boot config flash-filesystem:** *filename* **Example:** `Device(config)# boot config usbflash0:switch-config`	Specifies that the startup configuration file be stored in flash memory by setting the CONFIG_FILE variable.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 6	Do one of the following: • Use FTP, RCP, or TFTP to copy the new configuration. If you try to load a configuration that is more than three times larger than the NVRAM size, the following error message is displayed: "[buffer overflow - *file-size* /*buffer-size* bytes]. " • **configure terminal**	Enters the new configuration.

	Command or Action	Purpose
	Example: `Device# configure terminal`	
Step 7	**copy system:running-config nvram:startup-config** **Example:** `Device(config)# copy system:running-config nvram:startup-config`	When you have finished changing the running-configuration, save the new configuration.

Examples

The following example stores the configuration file in usbflash0:

```
Device# copy nvram:startup-config usbflash0:switch-config
 Device# configure terminal
 Device(config)# boot config usbflash0:switch-config
 Device(config)# end
 Device# copy system:running-config nvram:startup-config
```

Loading the Configuration Commands from the Network

To use a network server to store large configurations, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy system:running-config** {**ftp:** | **rcp:** | **tftp:**} **Example:** `Device# copy system:running-config ftp:`	Saves the running configuration to an FTP, RCP, or TFTP server.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**boot network** {**ftp:**[[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*] | **rcp:**[[[//[*username***@**]*location*]/*directory*]/*filename*] | **tftp:**[[[//*location*]/*directory*]/*filename*]} **Example:**	Specifies that the startup configuration file be loaded from the network server at startup.

	Command or Action	Purpose
	`Device(config)# boot network ftp://user1:guessme@example.com/dir10/file1`	
Step 5	**service config** **Example:** `Device(config)# service config`	Enables the switch to download configuration files at system startup.
Step 6	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 7	**copy system:running-config nvram:startup-config** **Example:** `Device# copy system:running-config nvram:startup-config`	Saves the configuration.

Copying Configuration Files from Flash Memory to the Startup or Running Configuration

To copy a configuration file from flash memory directly to your startup configuration in NVRAM or your running configuration, enter one of the commands in Step 2:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Do one of the following: • **copy** *filesystem*: [*partition-number***:**][*filename*] **nvram:startup-config** • **copy** *filesystem*: [*partition-number***:**][*filename*] **system:running-config** **Example:** `Device# copy usbflash0:4:ios-upgrade-1 nvram:startup-config`	• Loads a configuration file directly into NVRAM or • Copies a configuration file to your running configuration

Examples

The following example copies the file named ios-upgrade-1 from partition 4 of the flash memory PC Card in usbflash0 to the device startup configurations:

```
Device# copy usbflash0:4:ios-upgrade-1 nvram:startup-config

Copy 'ios-upgrade-1' from flash device as 'startup-config' ? [yes/no] yes

[OK]
```

Copying Configuration Files Between Flash Memory File Systems

On platforms with multiple flash memory file systems, you can copy files from one flash memory file system, such as internal flash memory to another flash memory file system. Copying files to different flash memory file systems lets you create backup copies of working configurations and duplicate configurations for other devices. To copy a configuration file between flash memory file systems, use the following commands in EXEC mode:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show** *source-filesystem*: **Example:** `Device# show flash:`	Displays the layout and contents of flash memory to verify the filename.
Step 3	**copy** *source-filesystem*: [*partition-number*:][*filename*] *dest-filesystem*:[*partition-number*:][*filename*] **Example:** `Device# copy flash: usbflash0:`	Copies a configuration file between flash memory devices. • The source device and the destination device cannot be the same. For example, the **copy usbflash0: usbflash0:** command is invalid.

Example

The following example copies the file named running-config from partition 1 on internal flash memory to partition 1 of usbflash0 on a device. In this example, the source partition is not specified, so the device prompts for the partition number:

```
Device# copy flash: usbflash0:

System flash
Partition   Size    Used      Free      Bank-Size  State          Copy Mode
```

```
  1         4096K   3070K     1025K     4096K      Read/Write     Direct
  2        16384K   1671K    14712K     8192K      Read/Write     Direct
[Type ?<no> for partition directory; ? for full directory; q to abort]
Which partition? [default = 1]
System flash directory, partition 1:
File  Length   Name/status
  1   3142748  dirt/network/mars-test/c3600-j-mz.latest
  2   850      running-config
[3143728 bytes used, 1050576 available, 4194304 total]
usbflash0 flash directory:
File  Length   Name/status
  1   1711088  dirt/gate/c3600-i-mz
  2   850      running-config
[1712068 bytes used, 2482236 available, 4194304 total]

Source file name? running-config
 Destination file name [running-config]?
Verifying checksum for 'running-config' (file # 2)...  OK
Erase flash device before writing? [confirm]
Flash contains files. Are you sure you want to erase? [confirm]
Copy 'running-config' from flash: device
  as 'running-config' into usbflash0: device WITH erase? [yes/no] yes

Erasing device... eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee ...erased!

 [OK - 850/4194304 bytes]
Flash device copy took 00:00:30 [hh:mm:ss]
Verifying checksum...  OK (0x16)
```

Copying a Configuration File from an FTP Server to Flash Memory Devices

To copy a configuration file from an FTP server to a flash memory device, complete the task in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	(Optional) Enters global configuration mode. This step is required only if you override the default remote username or password (see Steps 3 and 4).
Step 3	**ip ftp username** *username* **Example:** `Device(config)# ip ftp username Admin01`	(Optional) Specifies the remote username.
Step 4	**ip ftp password** *password* **Example:**	(Optional) Specifies the remote password.

	Command or Action	Purpose
	`Device(config)# ip ftp password adminpassword`	
Step 5	**end** **Example:** `Device(config)# end`	(Optional) Exits configuration mode. This step is required only if you override the default remote username (see Steps 3 and 4).
Step 6	**copy ftp:** [[*//location*]*/directory*]*/bundle_name* **flash:** **Example:** `Device>copy ftp:/ess9300_iosxe.17.04.01.SPA.bin flash:`	Copies the configuration file from a network server to the flash memory device using FTP.

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Copying a Configuration File from an RCP Server to Flash Memory Devices

To copy a configuration file from an RCP server to a flash memory device, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	(Optional) Enters global configuration mode. This step is required only if you override the default remote username or password (see Step 3).
Step 3	**ip rcmd remote-username** *username* **Example:** `Device(config)# ip rcmd remote-username Admin01`	(Optional) Specifies the remote username.
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Exits configuration mode. This step is required only if you override the default remote username or password (see Step 3).

	Command or Action	Purpose
Step 5	**copy rcp:** [[[//[*username@*]*location*]/*directory*] /*bundle_name*] **flash:** **Example:** `Device# copy rcp://netadmin@192.168.101.101/bundle1 flash:`	Copies the configuration file from a network server to the flash memory device using RCP. Respond to any device prompts for additional information or confirmation. Prompting depends on how much information you provide in the **copy** command and the current setting of the **file prompt** command.

Copying a Configuration File from a TFTP Server to Flash Memory Devices

To copy a configuration file from a TFTP server to a flash memory device, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy tftp:** [[[//*location*]/*directory*]/bundle_name **flash:** **Example:** `Device# copy tftp:/cat3k_caa-universalk9.SSA.03.12.02.EZP.150-12.02.EZP.150-12.02.EZP.bin flash:`	Copies the file from a TFTP server to the flash memory device. Reply to any device prompts for additional information or confirmation. Prompting depends on how much information you provide in the **copy** command and the current setting of the **file prompt** command.

Examples

The following example shows the copying of the configuration file named switch-config from a TFTP server to the flash memory card inserted in usbflash0. The copied file is renamed new-config.

```
Device#
copy tftp:switch-config usbflash0:new-config
```

Re-executing the Configuration Commands in the Startup Configuration File

To re-execute the commands located in the startup configuration file, complete the task in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure memory** **Example:** `Device# configure memory`	Re-executes the configuration commands located in the startup configuration file.

Clearing the Startup Configuration

You can clear the configuration information from the startup configuration. If you reboot the device with no startup configuration, the device enters the Setup command facility so that you can configure the device from scratch. To clear the contents of your startup configuration, complete the task in this section:

Important The IOS command parser may show a **factory-reset all** command. For embedded platforms this command is **NOT** supported as it leads to an ambiguity of which factory does it reference. A partner or integrator may install value add features that could be wiped out and not restored when such a command is executed. The system is obviously not in the state when it left the patrner or integrator's factory. If the desire is to perform a deep wipe of the on-board flash file system, the user should use the zeroization function and be completely familiar with the recovery features of the platform.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**erase nvram** **Example:**	Clears the contents of your startup configuration.

	Command or Action	Purpose
	`Device# erase nvram`	**Note** For all platforms except the Class A Flash file system platforms, this command erases NVRAM. The startup configuration file cannot be restored once it has been deleted. On Class A Flash file system platforms, when you use the **erase startup-config** EXEC command, the device erases or deletes the configuration pointed to by the CONFIG_FILE environment variable. If this variable points to NVRAM, the device erases NVRAM. If the CONFIG_FILE environment variable specifies a flash memory device and configuration filename, the device deletes the configuration file. That is, the device marks the file as "deleted," rather than erasing it. This feature allows you to recover a deleted file.

Deleting a Specified Configuration File

To delete a specified configuration on a specific flash device, complete the task in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**delete** *flash-filesystem***:***filename* **Example:** `Device# delete usbflash0:myconfig`	Deletes the specified configuration file on the specified flash device. **Note** On Class A and B Flash file systems, when you delete a specific file in flash memory, the system marks the file as deleted, allowing you to later recover a deleted file using the **undelete** EXEC command. Erased files cannot be recovered. To permanently erase the configuration file, use the **squeeze** EXEC command. On Class C Flash file systems, you cannot recover a file that has been deleted. If you attempt to erase or delete the configuration file specified by the CONFIG_FILE environment variable, the system prompts you to confirm the deletion.

Specifying the CONFIG_FILE Environment Variable on Class A Flash File Systems

On Class A flash file systems, you can configure the Cisco IOS software to load the startup configuration file specified by the CONFIG_FILE environment variable. The CONFIG_FILE variable defaults to NVRAM. To change the CONFIG_FILE environment variable, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy** [*flash-url* | *ftp-url* | *rcp-url* | *tftp-url* | **system:running-config** | **nvram:startup-config**] *dest-flash-url* **Example:** `Device# copy system:running-config nvram:startup-config`	Copies the configuration file to the flash file system from which the device loads the file on restart.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**boot config** *dest-flash-url* **Example:** `Device(config)# boot config 192.168.1.1`	Sets the CONFIG_FILE environment variable. This step modifies the runtime CONFIG_FILE environment variable.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 6	**copy system:running-config nvram:startup-config** **Example:** `Device# copy system:running-config nvram:startup-config`	Saves the configuration performed in Step 3 to the startup configuration.
Step 7	**show boot** **Example:** `Device# show boot`	(Optional) Allows you to verify the contents of the CONFIG_FILE environment variable.

Examples

The following example copies the running configuration file to the device. This configuration is then used as the startup configuration when the system is restarted:

```
Device# copy system:running-config usbflash0:config2
Device# configure terminal
Device(config)# boot config usbflash0:config2
Device(config)# end
Device# copy system:running-config nvram:startup-config
[ok]
Device# show boot
BOOT variable = usbflash0:rsp-boot-m
CONFIG_FILE variable = nvram:
Current CONFIG_FILE variable = usbflash0:config2
Configuration register is 0x010F
```

What to Do Next

After you specify a location for the startup configuration file, the **nvram:startup-config** command is aliased to the new location of the startup configuration file. The **more nvram:startup-config** EXEC command displays the startup configuration, regardless of its location. The **erase nvram:startup-config** EXEC command erases the contents of NVRAM and deletes the file pointed to by the CONFIG_FILE environment variable.

When you save the configuration using the **copy system:running-config nvram:startup-config** command, the device saves a complete version of the configuration file to the location specified by the CONFIG_FILE environment variable and a distilled version to NVRAM. A distilled version is one that does not contain access list information. If NVRAM contains a complete configuration file, the device prompts you to confirm your overwrite of the complete version with the distilled version. If NVRAM contains a distilled configuration, the device does not prompt you for confirmation and proceeds with overwriting the existing distilled configuration file in NVRAM.

Note If you specify a file in a flash device as the CONFIG_FILE environment variable, every time you save your configuration file with the **copy system:running-config nvram:startup-config** command, the old configuration file is marked as "deleted," and the new configuration file is saved to that device. Eventually, Flash memory fills up as the old configuration files still take up memory. Use the **squeeze** EXEC command to permanently delete the old configuration files and reclaim the space.

Configuring the Device to Download Configuration Files

You can specify an ordered list of network configuration and host configuration filenames. The Cisco IOS XE software scans this list until it loads the appropriate network or host configuration file.

To configure the device to download configuration files at system startup, perform at least one of the tasks described in the following sections:

- Configuring the Device to Download the Network Configuration File
- Configuring the Device to Download the Host Configuration File

If the device fails to load a configuration file during startup, it tries again every 10 minutes (the default setting) until a host provides the requested files. With each failed attempt, the device displays the following message on the console terminal:

```
Booting host-confg... [timed out]
```

If there are any problems with the startup configuration file, or if the configuration register is set to ignore NVRAM, the device enters the Setup command facility.

Configuring the Device to Download the Network Configuration File

To configure the Cisco IOS software to download a network configuration file from a server at startup, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**boot network** {**ftp:**[[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*] | **rcp:**[[[//[*username***@**]*location*]/*directory*]/*filename*] | **tftp:**[[[//*location*]/*directory*]/*filename*]} **Example:** `Device(config)# boot network tftp:hostfile1`	Specifies the network configuration file to download at startup, and the protocol to be used (TFTP, RCP, or FTP). • If you do not specify a network configuration filename, the Cisco IOS software uses the default filename network-confg. If you omit the address, the device uses the broadcast address. • You can specify more than one network configuration file. The software tries them in order entered until it loads one. This procedure can be useful for keeping files with different configuration information loaded on a network server.
Step 4	**service config** **Example:** `Device(config)# service config`	Enables the system to automatically load the network file on restart.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode.

	Command or Action	Purpose
Step 6	**copy system:running-config nvram:startup-config** **Example:** `Device# copy system:running-config nvram:startup-config`	Saves the running configuration to the startup configuration file.

Configuring the Device to Download the Host Configuration File

To configure the Cisco IOS software to download a host configuration file from a server at startup, complete the tasks in this section:

Procedure

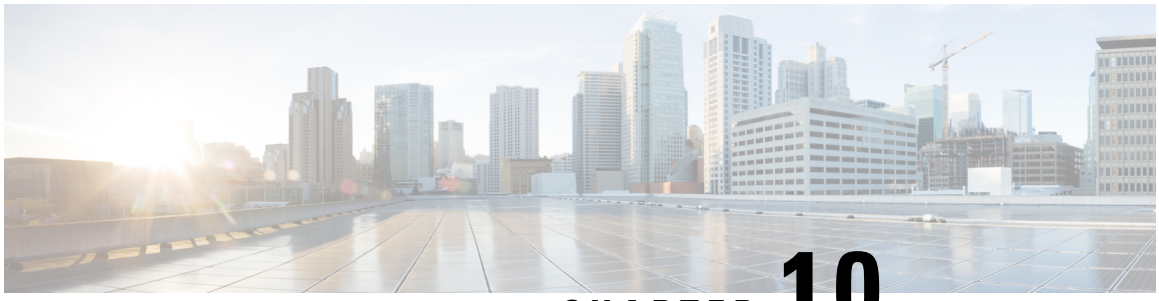
	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**boot host** {**ftp:**[[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*] | **rcp:**[[[//[*username***@**]*location*]/*directory*]/*filename*] | **tftp:**[[[//*location*]/*directory*]/*filename*] } **Example:** `Device(config)# boot host tftp:hostfile1`	Specifies the host configuration file to download at startup, and the protocol to be used (FTP, RCP, or TFTP): • If you do not specify a host configuration filename, the device uses its own name to form a host configuration filename by converting the name to all lowercase letters, removing all domain information, and appending "-confg." If no host name information is available, the software uses the default host configuration filename device-confg. If you omit the address, the device uses the broadcast address. • You can specify more than one host configuration file. The Cisco IOS software tries them in order entered until it loads one. This procedure can be useful for keeping files with different configuration information loaded on a network server.
Step 4	**service config** **Example:** `Device(config)# service config`	Enables the system to automatically load the host file upon restart.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 6	**copy system:running-config nvram:startup-config** **Example:** `Device# copy system:running-config nvram:startup-config`	Saves the running configuration to the startup configuration file.

Example

In the following example, a device is configured to download the host configuration file named hostfile1 and the network configuration file named networkfile1. The device uses TFTP and the broadcast address to obtain the file:

```
Device# configure terminal
Device(config)# boot host tftp:hostfile1
Device(config)# boot network tftp:networkfile1
Device(config)# service config
Device(config)# end
Device# copy system:running-config nvram:startup-config
```



CHAPTER 10

Secure Copy

This document provides the procedure to configure a Cisco device for Secure Copy (SCP) server-side functionality.

Prerequisites for Secure Copy

- Configure Secure Shell (SSH), authentication, and authorization on the device.
- Because the Secure Copy Protocol (SCP) relies on SSH for its secure transport, the device must have a Rivest, Shamir, and Adelman (RSA) key pair.

Information About Secure Copy

The Secure Copy feature provides a secure and authenticated method for copying switch configurations or switch image files. The Secure Copy Protocol (SCP) relies on Secure Shell (SSH), an application and a protocol that provides a secure replacement for the Berkeley r-tools.

The behavior of SCP is similar to that of Remote Copy Protocol (RCP), which comes from the Berkeley r-tools suite (Berkeley university's own set of networking applications), except that SCP relies on SSH for security. In addition, SCP requires authentication, authorization, and accounting (AAA) to be configured to ensure that the device can determine whether a user has the correct privilege level.

SCP allows only users with a privilege level of 15 to copy a file in the Cisco IOS File System (Cisco IFS) to and from a device by using the **copy** command. An authorized administrator can also perform this action from a workstation.

Note

- Enable the SCP option while using the `pscp.exe` file.
- An RSA public-private key pair must be configured on the device for SSH to work.

Secure Copy Performance Improvements

SSH bulk data transfer mode can be used to enhance the throughput performance of SCP that is operating in the capacity of a client or a server. Before Cisco IOS XE Dublin 17.10.1, this mode is disabled by default, but can be enabled by using the **ip ssh bulk-mode** global configuration command. Beginning from Cisco IOS XE Dublin 17.10.1, SSH bulk data transfer mode is enabled by default with a default window size of 128 KB. TCP selective acknowledgment (SACK) is enabled by default if the bulk mode window size is configured.

The default bulk mode window size of 128 KB is optimal to copy large files in most network settings. However, in long haul networks where the round-trip time (RTT) is high, 128 KB is not enough. You can enable the most optimal SCP throughput performance by configuring the bulk mode window size using the **ip ssh bulk-mode window-size** command. For example, in an ideal lab testing environment, a window size of 2 MB in a 200-milliseconds round-trip time (RTT) setting can give around 500 percent improved throughput performance when compared to the default 128 KB window size.

The bulk mode window size must be configured as per the network bandwidth-delay product, that is, a multiple of total available bandwidth in bits per second and the round-trip time (RTT) in seconds. Because the CPU usage may increase with the increased window size, make sure to balance this by choosing the right window size.

How to Configure Secure Copy

The following sections provide information about the Secure Copy configuration tasks.

Configuring Secure Copy

To configure a Cisco device for SCP server-side functionality, perform the following steps.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Sets AAA authentication at login.
Step 4	**aaa authentication login** {**default** | *list-name*} *method1* [*method2...*]	Enables the AAA access control system.

	Command or Action	Purpose
	Example: `Device(config)# aaa authentication login default group tacacs+`	
Step 5	**username** *name* [**privilege** *level*] **password** *encryption-type encrypted-password* **Example:** `Device(config)# username superuser privilege 2 password 0 superpassword`	Establishes a username-based authentication system. **Note** You can omit this step if a network-based authentication mechanism, such as TACACS+ or RADIUS, has been configured.
Step 6	**ip scp server enable** **Example:** `Device(config)# ip scp server enable`	Enables SCP server-side functionality.
Step 7	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 8	**debug ip scp** **Example:** `Device# debug ip scp`	(Optional) Troubleshoots SCP authentication problems.

Enabling Secure Copy on the SSH Server

The following task shows how to configure the server-side functionality for SCP. This task shows a typical configuration that allows a device to securely copy files from a remote workstation.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables the Authentication, Authorization, and Accounting (AAA) access control model.
Step 4	**aaa authentication login default local** **Example:** `Device(config)# aaa authentication login default local`	Sets AAA authentication to use the local username database for authentication at login.
Step 5	**aaa authorization exec default local** **Example:** `Device(config)# aaa authorization exec default local`	Sets the parameters that restrict user access to a network, runs the authorization to determine if the user ID is allowed to run an privileged EXEC shell, and specifies that the system must use the local database for authorization.
Step 6	**username** *name* **privilege** *privilege-level* **password** *password* **Example:** `Device(config)# username samplename privilege 15 password password1`	Establishes a username-based authentication system, and specifies the username, privilege level, and an unencrypted password. **Note** The minimum required value for the *privilege-level* argument is 15. A privilege level of less than 15 results in the connection closing.
Step 7	**ip ssh time-out** *seconds* **Example:** `Device(config)# ip ssh time-out 120`	Sets the time interval (in seconds) that the device waits for the SSH client to respond.
Step 8	**ip ssh authentication-retries** *integer* **Example:** `Device(config)# ip ssh authentication-retries 3`	Sets the number of authentication attempts after which the interface is reset.
Step 9	**ip scp server enable** **Example:** `Device(config)# ip scp server enable`	Enables the device to securely copy files from a remote workstation.
Step 10	**ip ssh bulk-mode** **Example:** `Device(config)# ip ssh bulk-mode`	(Optional) Enables SSH bulk data transfer mode to enhance the throughput performance of SCP.
Step 11	**exit** **Example:**	Exits global configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# exit`	
Step 12	**debug ip scp** **Example:** `Device# debug ip scp`	(Optional) Provides diagnostic information about SCP authentication problems.

Configuration Examples for Secure Copy

The following are examples of the Secure Copy configuration.

Example: Secure Copy Configuration Using Local Authentication

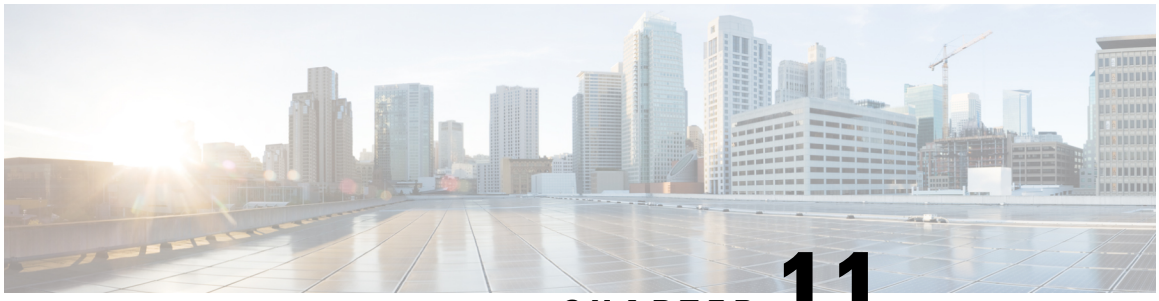
The following example shows how to configure the server-side functionality of Secure Copy. This example uses a locally defined username and password.

```
! AAA authentication and authorization must be configured properly in order for SCP to work.
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication login default local
Device(config)# aaa authorization exec default local
Device(config)# username user1 privilege 15 password 0 lab
! SSH must be configured and functioning properly.
Device(config)# ip scp server enable
Device(config)# end
```

Example: Secure Copy Server-Side Configuration Using Network-Based Authentication

The following example shows how to configure the server-side functionality of Secure Copy using a network-based authentication mechanism:

```
! AAA authentication and authorization must be configured properly for SCP to work.
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication login default group tacacs+
Device(config)# aaa authorization exec default group tacacs+
! SSH must be configured and functioning properly.
Device(config)# ip ssh time-out 120
Device(config)# ip ssh authentication-retries 3
Device(config)# ip scp server enable
Device(config)# end
```



CHAPTER 11

Configuration Replace and Configuration Rollback

Prerequisites for Configuration Replace and Configuration Rollback

The format of the configuration files used as input by the Configuration Replace and Configuration Rollback feature must comply with standard Cisco software configuration file indentation rules as follows:

- Start all commands on a new line with no indentation, unless the command is within a configuration submode.
- Indent commands within a first-level configuration submode one space.
- Indent commands within a second-level configuration submode two spaces.
- Indent commands within subsequent submodes accordingly.

These indentation rules describe how the software creates configuration files for such commands as **show running-config** or **copy running-config** *destination-url*. Any configuration file generated on a Cisco device complies with these rules.

Free memory larger than the combined size of the two configuration files (the current running configuration and the saved replacement configuration) is required.

Restrictions for Configuration Replace and Configuration Rollback

If the device does not have free memory larger than the combined size of the two configuration files (the current running configuration and the saved replacement configuration), the configuration replace operation is not performed.

Certain Cisco configuration commands such as those pertaining to physical components of a networking device (for example, physical interfaces) cannot be added or removed from the running configuration. For example, a configuration replace operation cannot remove the **interface ethernet 0** command line from the current running configuration if that interface is physically present on the device. Similarly, the **interface ethernet 1** command line cannot be added to the running configuration if no such interface is physically present on the device. A configuration replace operation that attempts to perform these types of changes results in error messages indicating that these specific command lines failed.

In very rare cases, certain Cisco configuration commands cannot be removed from the running configuration without reloading the device. A configuration replace operation that attempts to remove this type of command results in error messages indicating that these specific command lines failed.

Information About Configuration Replace and Configuration Rollback

Configuration Archive

The Cisco IOS configuration archive is intended to provide a mechanism to store, organize, and manage an archive of Cisco IOS configuration files to enhance the configuration rollback capability provided by the **configure replace** command. Before this feature was introduced, you could save copies of the running configuration using the **copy running-config** *destination-url* command, storing the replacement file either locally or remotely. However, this method lacked any automated file management. On the other hand, the Configuration Replace and Configuration Rollback feature provides the capability to automatically save copies of the running configuration to the Cisco IOS configuration archive. These archived files serve as checkpoint configuration references and can be used by the **configure replace** command to revert to previous configuration states.

The **archive config** command allows you to save Cisco IOS configurations in the configuration archive using a standard location and filename prefix that is automatically appended with an incremental version number (and optional timestamp) as each consecutive file is saved. This functionality provides a means for consistent identification of saved Cisco IOS configuration files. You can specify how many versions of the running configuration are kept in the archive. After the maximum number of files are saved in the archive, the oldest file is automatically deleted when the next, most recent file is saved. The **show archive** command displays information for all configuration files saved in the Cisco IOS configuration archive.

The Cisco IOS configuration archive, in which the configuration files are stored and available for use with the **configure replace** command, can be located on the following file systems: FTP, HTTP, RCP, TFTP.

Configuration Replace

The **configure replace** privileged EXEC command provides the capability to replace the current running configuration with any saved Cisco IOS configuration file. This functionality can be used to revert to a previous configuration state, effectively rolling back any configuration changes that were made since the previous configuration state was saved.

When using the **configure replace** command, you must specify a saved Cisco IOS configuration as the replacement configuration file for the current running configuration. The replacement file must be a complete configuration generated by a Cisco IOS device (for example, a configuration generated by the **copy running-config** *destination-url* command), or, if generated externally, the replacement file must comply with the format of files generated by Cisco IOS devices. When the **configure replace** command is entered, the current running configuration is compared with the specified replacement configuration and a set of diffs is generated. The algorithm used to compare the two files is the same as that employed by the **show archive config differences** command. The resulting diffs are then applied by the Cisco IOS parser to achieve the replacement configuration state. Only the diffs are applied, avoiding potential service disruption from reapplying configuration commands that already exist in the current running configuration. This algorithm effectively handles configuration changes to order-dependent commands (such as access lists) through a multiple pass process. Under normal circumstances, no more than three passes are needed to complete a configuration replace operation, and a limit of five passes is performed to preclude any looping behavior.

The Cisco IOS **copy** *source-url* **running-config** privileged EXEC command is often used to copy a stored Cisco IOS configuration file to the running configuration. When using the **copy** *source-url* **running-config** command as an alternative to the **configure replace** *target-url* privileged EXEC command, the following major differences should be noted:

- The **copy** *source-url* **running-config** command is a merge operation and preserves all of the commands from both the source file and the current running configuration. This command does not remove commands from the current running configuration that are not present in the source file. In contrast, the **configure replace** *target-url* command removes commands from the current running configuration that are not present in the replacement file and adds commands to the current running configuration that need to be added.
- The **copy** *source-url* **running-config** command applies every command in the source file, whether or not the command is already present in the current running configuration. This algorithm is inefficient and, in some cases, can result in service outages. In contrast, the **configure replace** *target-url* command only applies the commands that need to be applied—no existing commands in the current running configuration are reapplied.
- A partial configuration file may be used as the source file for the **copy** *source-url* **running-config** command, whereas a complete Cisco IOS configuration file must be used as the replacement file for the **configure replace** *target-url* command.

A locking feature for the configuration replace operation was introduced. When the **configure replace** command is used, the running configuration file is locked by default for the duration of the configuration replace operation. This locking mechanism prevents other users from changing the running configuration while the replacement operation is taking place, which might otherwise cause the replacement operation to terminate unsuccessfully. You can disable the locking of the running configuration by using the **no lock** keyword when issuing the **configure replace** command.

The running configuration lock is automatically cleared at the end of the configuration replace operation. You can display any locks that may be currently applied to the running configuration using the **show configuration lock** command.

Configuration Rollback

The concept of rollback comes from the transactional processing model common to database operations. In a database transaction, you might make a set of changes to a given database table. You then must choose whether to commit the changes (apply the changes permanently) or to roll back the changes (discard the changes and revert to the previous state of the table). In this context, rollback means that a journal file containing a log of the changes is discarded, and no changes are applied. The result of the rollback operation is to revert to the previous state, before any changes were applied.

The **configure replace** command allows you to revert to a previous configuration state, effectively rolling back changes that were made since the previous configuration state was saved. Instead of basing the rollback operation on a specific set of changes that were applied, the Cisco IOS configuration rollback capability uses the concept of reverting to a specific configuration state based on a saved Cisco IOS configuration file. This concept is similar to the database idea of saving a checkpoint (a saved version of the database) to preserve a specific state.

If the configuration rollback capability is desired, you must save the Cisco IOS running configuration before making any configuration changes. Then, after entering configuration changes, you can use that saved configuration file to roll back the changes (using the **configure replace** *target-url* command). Furthermore, because you can specify any saved Cisco IOS configuration file as the replacement configuration, you are not limited to a fixed number of rollbacks, as is the case in some rollback models.

Configuration Rollback Confirmed Change

The Configuration Rollback Confirmed Change feature allows configuration changes to be performed with an optional requirement that they be confirmed. If this confirmation is not received, the configuration is returned to the state prior to the changes being applied. The mechanism provides a safeguard against inadvertent loss of connectivity between a network device and the user or management application due to configuration changes.

Benefits of Configuration Replace and Configuration Rollback

- Allows you to revert to a previous configuration state, effectively rolling back configuration changes.
- Allows you to replace the current running configuration file with the startup configuration file without having to reload the device or manually undo CLI changes to the running configuration file, therefore reducing system downtime.
- Allows you to revert to any saved Cisco IOS configuration state.
- Simplifies configuration changes by allowing you to apply a complete configuration file to the device, where only the commands that need to be added or removed are affected.
- When using the **configure replace** command as an alternative to the **copy** *source-url* **running-config** command, increases efficiency and prevents risk of service outages by not reapplying existing commands in the current running configuration.

How to Use Configuration Replace and Configuration Rollback

Creating a Configuration Archive

No prerequisite configuration is needed to use the **configure replace** command. Using the **configure replace** command in conjunction with the Cisco IOS configuration archive and the **archive config** command is optional but offers significant benefit for configuration rollback scenarios. Before using the **archive config** command, the configuration archive must be configured. Perform this task to configure the characteristics of the configuration archive.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**archive** **Example:** `Device(config)# archive`	Enters archive configuration mode.
Step 4	**path** *url* **Example:** `Device(config-archive)# path flash:myconfiguration`	Specifies the location and filename prefix for the files in the Cisco IOS configuration archive. **Note** If a directory is specified in the path instead of file, the directory name must be followed by a forward slash as follows: path flash:/directory/. The forward slash is not necessary after a filename; it is only necessary when specifying a directory.
Step 5	**maximum** *number* **Example:** `Device(config-archive)# maximum 14`	(Optional) Sets the maximum number of archive files of the running configuration to be saved in the Cisco IOS configuration archive. • The *number* argument is the maximum number of archive files of the running configuration to be saved in the Cisco IOS configuration archive. Valid values are from 1 to 14. The default is 10.

	Command or Action	Purpose
		Note Before using this command, you must configure the **path** command to specify the location and filename prefix for the files in the Cisco IOS configuration archive.
Step 6	**time-period** *minutes* **Example:** `Device(config-archive)# time-period 1440`	(Optional) Sets the time increment for automatically saving an archive file of the current running configuration in the Cisco IOS configuration archive. • The *minutes* argument specifies how often, in minutes, to automatically save an archive file of the current running configuration in the Cisco IOS configuration archive. **Note** Before using this command, you must configure the **path** command to specify the location and filename prefix for the files in the Cisco IOS configuration archive.
Step 7	**end** **Example:** `Device(config-archive)# end`	Exits to privileged EXEC mode.
Step 8	**archive config** **Example:** `Device# archive config`	Saves the current running configuration file to the configuration archive. **Note** The **path** command must be configured before using this command.

Performing a Configuration Replace or Configuration Rollback Operation

Perform this task to replace the current running configuration file with a saved Cisco IOS configuration file.

Note You must create a configuration archive before performing this procedure. See Creating a Configuration Archive for detailed steps. The following procedure details how to return to that archived configuration in the event of a problem with the current running configuration.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose

Step 2	**configure replace** *target-url* [**nolock**] [**list**] [**force**] [**ignore case**] [**revert trigger** [**error**] [**timer** *minutes*] | **time** *minutes*]]
	Example:
	`Device# configure replace flash: startup-config time 120`
		Replaces the current running configuration file with a saved Cisco IOS configuration file.
		• The *target - url* argument is a URL (accessible by the Cisco IOS file system) of the saved Cisco IOS configuration file that is to replace the current running configuration, such as the configuration file created using the **archive config** command.
		• The **list** keyword displays a list of the command lines applied by the Cisco IOS software parser during each pass of the configuration replace operation. The total number of passes performed is also displayed.
		• The **force** keyword replaces the current running configuration file with the specified saved Cisco IOS configuration file without prompting you for confirmation.
		• The **time** *minutes* keyword and argument specify the time (in minutes) within which you must enter the **configure confirm** command to confirm replacement of the current running configuration file. If the **configure confirm** command is not entered within the specified time limit, the configuration replace operation is automatically reversed (in other words, the current running configuration file is restored to the configuration state that existed prior to entering the **configure replace** command).
		• The **nolock** keyword disables the locking of the running configuration file that prevents other users from changing the running configuration during a configuration replace operation.
		• The **revert trigger** keywords set the following triggers for reverting to the original configuration:
		  • **error** —Reverts to the original configuration upon error.
		  • **timer** *minutes* —Reverts to the original configuration if specified time elapses.
		• The **ignore case** keyword allows the configuration to ignore the case of the confirmation command.

Step 3	**configure revert** { **now** | **timer** {*minutes* | **idle** *minutes*} } **Example:** `Device# configure revert now`	(Optional) To cancel the timed rollback and trigger the rollback immediately, or to reset parameters for the timed rollback, use the **configure revert**command in privileged EXEC mode. • **now** —Triggers the rollback immediately.

	Command or Action	Purpose
		• **timer** —Resets the configuration revert timer. • Use the *minutes* argument with the **timer** keyword to specify a new revert time in minutes. • Use the **idle** keyword along with a time in minutes to set the maximum allowable time period of no activity before reverting to the saved configuration.
Step 4	**configure confirm** **Example:** `Device# configure confirm`	(Optional) Confirms replacement of the current running configuration file with a saved Cisco IOS configuration file. **Note** Use this command only if the **time** *seconds* keyword and argument of the **configure replace** command are specified.
Step 5	**exit** **Example:** `Device# exit`	Exits to user EXEC mode.

Monitoring and Troubleshooting the Feature

Perform this task to monitor and troubleshoot the Configuration Replace and Configuration Rollback feature.

Step 1 **enable**

Use this command to enable privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
Device#
```

Step 2 **show archive**

Use this command to display information about the files saved in the Cisco IOS configuration archive.

Example:

```
Device# show archive
There are currently 1 archive configurations saved.
The next archive file will be named flash:myconfiguration-2
 Archive #  Name
   0
   1       flash:myconfiguration-1 <- Most Recent
   2
   3
   4
   5
```

```
   6
   7
   8
   9
   10
   11
   12
   13
   14
```

The following is sample output from the **show archive** command after several archive files of the running configuration have been saved. In this example, the maximum number of archive files to be saved is set to three.

Example:

```
Device# show archive
There are currently 3 archive configurations saved.
The next archive file will be named flash:myconfiguration-8
 Archive #  Name
   0
   1       :Deleted
   2       :Deleted
   3       :Deleted
   4       :Deleted
   5       flash:myconfiguration-5
   6       flash:myconfiguration-6
   7       flash:myconfiguration-7 <- Most Recent
   8
   9
   10
   11
   12
   13
   14
```

Step 3 debug archive versioning

Use this command to enable debugging of the Cisco IOS configuration archive activities to help monitor and troubleshoot configuration replace and rollback.

Example:

```
Device# debug archive versioning
Jan  9 06:46:28.419:backup_running_config
Jan  9 06:46:28.419:Current = 7
Jan  9 06:46:28.443:Writing backup file flash:myconfiguration-7
Jan  9 06:46:29.547: backup worked
```

Step 4 debug archive config timestamp

Use this command to enable debugging of the processing time for each integral step of a configuration replace operation and the size of the configuration files being handled.

Example:

```
Device# debug archive config timestamp
Device# configure replace flash:myconfiguration force
Timing Debug Statistics for IOS Config Replace operation:
       Time to read file usbflash0:sample_2.cfg = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
Starting Pass 1
```

```
        Time to read file system:running-config = 0 msec (0 sec)
        Number of lines read:93
        Size of file        :2539
        Time taken for positive rollback pass = 320 msec (0 sec)
        Time taken for negative rollback pass = 0 msec (0 sec)
        Time taken for negative incremental diffs pass = 59 msec (0 sec)
        Time taken by PI to apply changes = 0 msec (0 sec)
        Time taken for Pass 1 = 380 msec (0 sec)
Starting Pass 2
        Time to read file system:running-config = 0 msec (0 sec)
        Number of lines read:55
        Size of file        :1054
        Time taken for positive rollback pass = 0 msec (0 sec)
        Time taken for negative rollback pass = 0 msec (0 sec)
        Time taken for Pass 2 = 0 msec (0 sec)
Total number of passes:1
Rollback Done
```

Step 5 **exit**

Use this command to exit to user EXEC mode.

Example:

```
Device# exit
Device>
```

Configuration Examples for Configuration Replace and Configuration Rollback

Creating a Configuration Archive

The following example shows how to perform the initial configuration of the Cisco IOS configuration archive. In this example, flash:myconfiguration is specified as the location and filename prefix for the files in the configuration archive and a value of 10 is set as the maximum number of archive files to be saved.

```
configure terminal
!
archive
 path flash:myconfiguration
 maximum 10
end
```

Replacing the Current Running Configuration with a Saved Cisco IOS Configuration File

The following example shows how to replace the current running configuration with a saved Cisco IOS configuration file named flash:myconfiguration. The **configure replace** command interactively prompts you to confirm the operation.

```
Device# configure replace flash:myconfiguration
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
Total number of passes: 1
Rollback Done
```

In the following example, the **list** keyword is specified in order to display the command lines that were applied during the configuration replace operation:

```
Device# configure replace flash:myconfiguration list
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
!Pass 1
!List of Commands:
no snmp-server community public ro
snmp-server community mystring ro

end
Total number of passes: 1
Rollback Done
```

Reverting to the Startup Configuration File

The following example shows how to revert to the Cisco IOS startup configuration file using the **configure replace** command. This example also shows the use of the optional **force** keyword to override the interactive user prompt:

```
Device# configure replace flash:startup-config force
Total number of passes: 1
Rollback Done
```

Performing a Configuration Replace Operation with the configure confirm Command

The following example shows the use of the **configure replace** command with the **time** *minutes* keyword and argument. You must enter the **configure confirm** command within the specified time limit to confirm replacement of the current running configuration file. If the **configure confirm** command is not entered within the specified time limit, the configuration replace operation is automatically reversed (in other words, the current running configuration file is restored to the configuration state that existed prior to entering the **configure replace** command).

```
Device# configure replace flash:startup-config time 120
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
Total number of passes: 1
```

```
Rollback Done
Device# configure confirm
```

The following example shows the use of the **configure revert** command with the **timer** keyword. You must enter the **configure revert** command to cancel the timed rollback and trigger the rollback immediately, or to reset parameters for the timed rollback.

```
Device# configure revert timer 100
```

Performing a Configuration Rollback Operation

The following example shows how to make changes to the current running configuration and then roll back the changes. As part of the configuration rollback operation, you must save the current running configuration before making changes to the file. In this example, the **archive config** command is used to save the current running configuration. The generated output of the **configure replace** command indicates that only one pass was performed to complete the rollback operation.

Note Before using the **archive config** command, you must configure the **path** command to specify the location and filename prefix for the files in the Cisco IOS configuration archive.

You first save the current running configuration in the configuration archive as follows:

```
archive config
```

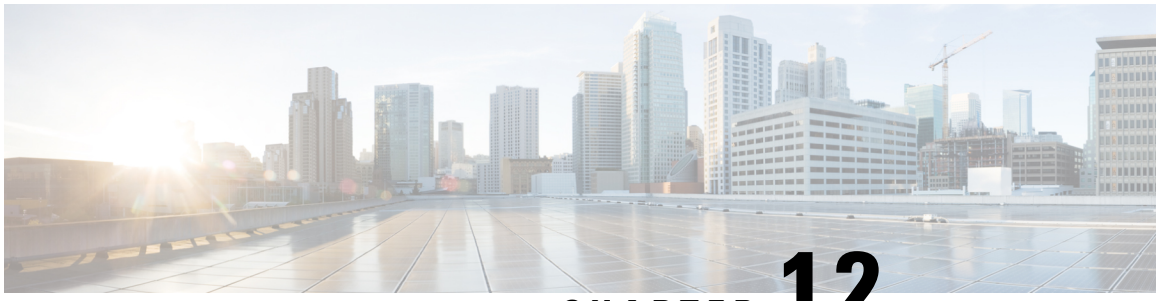
You then enter configuration changes as shown in the following example:

```
configure terminal
!
user netops2 password rain
user netops3 password snow
exit
```

After having made changes to the running configuration file, assume you now want to roll back these changes and revert to the configuration that existed before the changes were made. The **show archive** command is used to verify the version of the configuration to be used as a replacement file. The **configure replace** command is then used to revert to the replacement configuration file as shown in the following example:

```
Device# show archive
There are currently 1 archive configurations saved.
The next archive file will be named flash:myconfiguration-2
 Archive #  Name
   0
   1       flash:myconfiguration-1 <- Most Recent
   2
   3
   4
   5
   6
   7
   8
   9
```

```
   10
Device# configure replace flash:myconfiguration-1
Total number of passes: 1
Rollback Done
```



CHAPTER 12

Software Maintenance Upgrade

The Software Maintenance Upgrade (SMU) is a package that can be installed on a system to provide a fix or a security resolution to a released image.

Information About Software Maintenance Upgrade

SMU Overview

The SMU is a package that can be installed on a system to provide a fix or a security resolution to a released image. An SMU package is provided on a per release and per component basis.

An SMU provides a significant benefit over classic Cisco IOS software because it allows you to address network issues quickly while reducing the time and scope of the testing required. The Cisco IOS XE platform internally validates SMU compatibility and does not allow you to install noncompatible SMUs.

All the SMUs are integrated into the subsequent Cisco IOS XE software maintenance releases. An SMU is an independent and self-sufficient package and it does not have any prerequisites or dependencies. You can choose which SMUs to install or uninstall in any order.

SMUs are supported only on Extended Maintenance releases and for the full lifecycle of the underlying software release.

Perform these basic steps to install an SMU:

1. Add the SMU to the filesystem.
2. Activate the SMU on the system.
3. Commit the SMU changes so that it is persistent across reloads.

SMU Workflow

The SMU process is initiated with a request to the Cisco Customer Support. Contact your customer support to raise an SMU request.

At release time, the SMU package is posted to the Cisco Software Download page and can be downloaded and installed.

SMU Package

The SMU package contains a small set of files for patching the release along with metadata that describes the contents of the package, and fix for the reported issue that the SMU is requested for. The SMU package also supports patching of the public key infrastructure (PKI) component.

SMU Reload

All SMUs require a cold reload of the system during activation. A cold reload is the complete reload of the operating system. This action affects the traffic flow for the duration of the reload. This reload ensures that all processes are started with the correct libraries and files that are installed as part of the SMU.

How to Manage Software Maintenance Updates

You can install, activate, and commit an SMU package using a single command (1-step process) or using separate commands (3-step process).

Tip Use the 1-step process when you have to install just one SMU package file and use the 3-step process when you have to install multiple SMUs. The 3-step process minimises the number of reloads required when you have more than one SMU package file to install.

Installing an SMU Package

This task shows how to use the **install add file activate commit** command for installing an SMU package.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**install add file flash:** *filename* [**activate commit**] **Example:** `Device# install add file flash:ess9300_iosxe.BLD_SMU_20180302_085005_TWIG_LATEST_20180306_013805.3.SSA.smu.bin activate commit`	Copies the maintenance update package from flash, performs a compatibility check for the platform and image versions, activates the SMU package, and makes the package persistent across reloads. This command extracts the individual components of the .bin file into the subpackages and packages.conf files. You can also copy the maintenance update package from from a remote location (through FTP, HTTP, HTTPS, or TFTP).

	Command or Action	Purpose
		Note If the SMU file is copied using TFTP, use bootflash to activate the SMU.
Step 3	**exit** **Example:** `Device# exit`	Exits privileged EXEC mode and returns to user EXEC mode.

Managing an SMU Package

SUMMARY STEPS

1. **enable**
2. **install add file flash:** *filename*
3. **install activate file flash:** *filename*
4. **install commit**
5. **install rollback to** {**base** | **committed** | **id** *commit-ID*}
6. **install deactivate file flash:** *filename*
7. **install remove** {**file flash:** *filename* | **inactive**}
8. **show version**
9. **show install summary**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**install add file flash:** *filename* **Example:** `Device# install add file flash:ess9300_iosxe.BLD_SMU_20180302_085005_ TWIG_LATEST_20180306_013805.3.SSA.smu.bin`	Copies the SMU package from a source location to the device (in case source location is remote), and then performs a compatibility check for the platform and image versions, and adds the SMU package on all member nodes or FRUs, as applicable. This command also runs base compatibility checks on a file to ensure that the SMU package is supported on the platform. It also adds an entry in the `package/SMU.sta` file, so that its status can be monitored and maintained.
Step 3	**install activate file flash:** *filename* **Example:** `Device# install activate add file flash:ess9300_iosxe.BLD_SMU_20180302_085005_ TWIG_LATEST_20180306_013805.3.SSA.smu.bin`	Runs compatibility checks, installs the package, and updates the package status details.

	Command or Action	Purpose
Step 4	**install commit** **Example:** `Device# install commit`	Commits the activation changes to be persistent across reloads. The commit can be done after activation when the system is up, or after the first reload. If a package is activated, but not committed, it remains active after the first reload, but not after the second reload.
Step 5	**install rollback to** {**base** | **committed** | **id** *commit-ID*} **Example:** `Device# install rollback to committed`	Returns the device to the previous installation state.
Step 6	**install deactivate file flash:** *filename* **Example:** `Device# install deactivate file flash:ess9300_iosxe.BLD_SMU_20180302_085005_TWIG_LATEST_20180306_013805.3.SSA.smu.bin`	Deactivates an active package and updates the package status.
Step 7	**install remove** {**file flash:** *filename* | **inactive**} **Example:** `Device# install remove file flash:ess9300_iosxe.BLD_SMU_20180302_085005_TWIG_LATEST_20180306_013805.3.SSA.smu.bin`	Verifies if the specified SMU is inactive and if it is, deletes it from the file system. The **inactive** option deletes all the inactive packages from the file system.
Step 8	**show version** **Example:** `Device# show version`	Displays the image version on the device.
Step 9	**show install summary** **Example:** `Device# show install summary`	Displays information about the installation status of packages. The output of this command varies according to the **install** commands that are configured.

Installing an SMU Package: 1-Step Process

This task shows how to use the single **install add file activate commit** command for installing an SMU package.

Before you begin

Check that the SMU you are about to install corresponds to the software image installed on your device. For example, SMU `ess9300_lite_iosxe.17.04.01.CSCvk70181.SPA.smu.bin` is compatible with software image `ess9300_lite_iosxe.17.04.01.SPA.bin`.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. Enter your password, if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**install add file flash:** *filename* [**activate commit**] **Example:** `Device# install add file flash:ess9300_lite_iosxe.17.04.01.CSCvk70181.SPA.smu.bin activate commit`	Copies the maintenance update package from flash to the device, performs a compatibility check for the platform and image versions, activates the SMU package, and makes the package persistent across reloads. This command extracts the individual components of the .bin file into the subpackages and packages.conf files. You can also copy the SMU package from from a remote location (through FTP, HTTP, HTTPS, or TFTP). **Note** If the SMU file is copied using TFTP, use bootflash to activate the SMU.
Step 3	**exit** **Example:** `Device# exit`	Exits privileged EXEC mode and returns to user EXEC mode.

Managing an SMU

This task shows how to rollback the installation state, deactivate, and remove a previously installed SMU package from the device. This can be used for a SMU that has been installed with the 1-step and 3-step process.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**install rollback to** {**base** | **committed** | **id** *commit-ID*} **Example:** `Device# install rollback to committed`	Returns the device to the previous installation state. After the rollback, a reload is required.
Step 3	**install deactivate file** *location filename* **Example:** `Device# install deactivate file flash:ess9300_lite_iosxe.17.04.04.CSCvk70181.SPA.smu.bin`	Deactivates an active package, updates the package status, and triggers a process to restart or reload.
Step 4	**install remove** {**file** *location filename* | **inactive**} **Example:** `Device# install remove file flash:ess9300_lite_iosxe.17.04.04.CSCvk70181.SPA.smu.bin`	Checks if the specified SMU is inactive and if it is, deletes it from the file system. The **inactive** option deletes all the inactive packages from the file system.
Step 5	**show version** **Example:**	Displays the image version on the device.

	Command or Action	Purpose
	`Device# show version`	
Step 6	**show install summary** **Example:** `Device# show install summary`	Displays information about the active package. The output of this command varies according to the **install** commands that are configured.

Configuration Examples for Software Maintenance Upgrade

The following is a list of SMU configuration examples.

- Example: Installing an SMU (3-Step Process, Using flash:), on page 184
- Example: Installing Multiple SMUs (3-Step Process, Using flash:), on page 187
- Example: Installing an SMU (3-Step Process, Using TFTP), on page 192
- Example: Managing a SMU Package (Additional show commands, Rollback, Deactivation), on page 194

Example: Installing an SMU (3-Step Process, Using flash:)

The following example shows how to install a SMU package by using the 3-step process. Here the SMU package file is saved in the device's flash.

1. Copying the SMU package file from flash and installing it.

```
Device# install add file flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
install_add: START Wed Jun 10 14:17:45 IST 2020
install_add: Adding SMU

--- Starting initial file syncing ---
Info: Finished copying flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin to the
selected switch(es)
Finished initial file syncing


*Jun 10 14:17:48.128 IST: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
 Started install add flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.binExecuting pre
 scripts....
Executing pre sripts done.
--- Starting SMU Add operation ---
Performing SMU_ADD on all members
  [1] SMU_ADD package(s) on switch 1
  [1] Finished SMU_ADD on switch 1
Checking status of SMU_ADD on [1]
SMU_ADD: Passed on [1]
Finished SMU Add operation

SUCCESS: install_add /flash/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin Wed Jun 10
 14:18:00 IST 2020
```

Verifying the addition and installation of the SMU package file by using the **show install summary** command. The status of the SMU package file is `I`, because it has not been activated and committed yet.

```
Device# show install summary

[ Switch 1 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
IMG   C    16.9.4.0.3431

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

2. Activating the SMU package file.

```
Device# install activate file flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

install_activate: START Wed Jun 10 14:19:59 IST 2020
install_activate: Activating SMU

*Jun 10 14:20:01.513 IST: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
 Started install activate flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

This operation requires a reload of the system. Do you want to proceed? [y/n]y
Executing pre scripts....
Executing pre sripts done.

--- Starting SMU Activate operation ---
Performing SMU_ACTIVATE on all members
  [1] SMU_ACTIVATE package(s) on switch 1
  [1] Finished SMU_ACTIVATE on switch 1
Checking status of SMU_ACTIVATE on [1]
SMU_ACTIVATE: Passed on [1]
Finished SMU Activate operation

install_activate: Reloading the box to complete activation of the SMU...
install_activate will reload the system now!

*Jun 10 14:20:22.258 IST: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 1 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds
     Chassis 1 reloading, reason - Reload command
Jun 10 14:20:28.291: %PMAN-5-EXITACTION: F0/0: pvp: Process manager is exiting: reload
fp action requested
Jun 10 14:20:30.718: %PMAN-5-EXITACTION: R0/0: pvp: Proce
Jun 10 14:20:34.834: %PMAN-5-EXITACTION: C0/0: pvp: Process manager is exiting:
Jun 10 14:20:36.053: %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
 install activate SMU flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
watchdog watchdog0: watchdog did not stop!
reboot: Restarting system

Initializing Hardware...
<output truncated>

############
Jun 10 08:52:01.806: %BOOT-5-BOOTTIME_SMU_TEMP_ACTIVE_DETECTED: R0/0: install_engine:
SMU file /flash/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin active temporary...
SMU commit is pending

Cisco IOS Software [Fuji], Catalyst L3 Switch Software (CAT9K_LITE_IOSXE), Version 16.9.4,
 RELEASE SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
```

```
Copyright (c) 1986-2019 by Cisco Systems, Inc.
Compiled Thu 22-Aug-19 17:30 by mcpre

<output truncated>
```

Verifying activation of the SMU package file by using the **show install summary** command. The status of the SMU package file is U, because it has not been committed yet.

```
[ Switch 1 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   U    flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
IMG   C    16.9.4.0.3431

--------------------------------------------------------------------------------
Auto abort timer: active on install_activate, time before rollback - 01:41:52
--------------------------------------------------------------------------------
```

3. Committing the SMU package file

```
Device# install commit
install_commit: START Wed Jun 10 14:38:42 IST 2020
install_commit: Committing SMU

*Jun 10 14:38:44.906 IST: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
 Started install commitExecuting pre scripts....
Executing pre sripts done.
--- Starting SMU Commit operation ---
Performing SMU_COMMIT on all members
  [1] SMU_COMMIT package(s) on switch 1
  [1] Finished SMU_COMMIT on switch 1
Checking status of SMU_COMMIT on [1]
SMU_COMMIT: Passed on [1]
Finished SMU Commit operation

SUCCESS: install_commit /flash/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin Wed Jun
 10 14:38:58 IST 2020
*Jun 10 14:38:59.385 IST: %INSTALL-5-INSTALL_COMPLETED_INFO: Switch 1 R0/0:
install_engine: Completed install commit SMU
```

Verifying the commit by using the **show install summary** command. The SMU package file has been installed, activated and committed and the status is C.

```
Device# show install summary
[ Switch 1 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   C    flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
IMG   C    16.9.4.0.3431

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

Verifying active packages by using the **show install active** command

```
Device# show install active
[ Switch 1 ] Active Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   C    flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
IMG   C    16.9.4.0.3431
```

Checking the version, by using the **show version** command:

```
Device# show version
Cisco IOS XE Software, Version 16.09.04
Cisco IOS Software [Fuji], Catalyst L3 Switch Software (CAT9K_LITE_IOSXE), Version 16.9.4,
 RELEASE SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2019 by Cisco Systems, Inc.
Compiled Thu 22-Aug-19 17:30 by mcpre
...
```

Example: Installing Multiple SMUs (3-Step Process, Using flash:)

The following example shows how to install multiple SMU package files by using the 3-step process. Here the SMU package files are saved in the device's flash.

The SMU files being installed on the switch stack are:
`cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin` and
`cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin`

1. (Optional) Checking that the switch stack is ready and that the SMU package files are in the device's flash.

```
Device# show switch
Switch/Stack Mac Address : 08ec.f586.aa80 - Local Mac Address
Mac persistency wait time: Indefinite
                                             H/W   Current
Switch#   Role    Mac Address     Priority Version  State
-----------------------------------------------------------------------------------
*1       Active   08ec.f586.aa80     1      V01     Ready
 2       Member   7488.bb3c.f600     1      V01     Ready
 3       Member   7488.bb3f.9c00     1      V01     Ready
 4       Member   08ec.f5ee.1080     1      V01     Ready
 5       Standby  08ec.f589.7c80     1      V01     Ready

Device# dir flash: | i smu

89075  -rw-  79256 Oct 26 2035 07:07:42 +00:00
cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
89082  -rw-  9656  Oct 26 2035 07:08:08 +00:00
cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
```

2. Copying the SMU package files from flash and adding them.

   Only one SMU package file is added at a time; no reload is required between the addition of the SMU package files.

```
Device# install add file flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
install_add: START Fri Oct 26 07:10:59 UTC 2035
Oct 26 07:11:01.695 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
```

```
 add flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
install_add: Adding SMU
install_add: Checking whether new add is allowed ....

--- Starting initial file syncing ---

*Oct 26 07:11:01.643: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
Started install add flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin[1]: Copying
flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin from switch 1 to switch 2 3 4 5
[2 3 4 5]: Finished copying to switch 2 switch 3 switch 4 switch 5
Info: Finished copying flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin to the
selected switch(es)
Finished initial file syncing

--- Starting SMU Add operation ---
Performing SMU_ADD on all members
  [1] SMU_ADD package(s) on switch 1
  [1] Finished SMU_ADD on switch 1
  [2] SMU_ADD package(s) on switch 2
  [2] Finished SMU_ADD on switch 2
  [3] SMU_ADD package(s) on switch 3
  [3] Finished SMU_ADD on switch 3
  [4] SMU_ADD package(s) on switch 4
  [4] Finished SMU_ADD on switch 4
  [5] SMU_ADD package(s) on switch 5
  [5] Finished SMU_ADD on switch 5
Checking status of SMU_ADD on [1 2 3 4 5]
SMU_ADD: Passed on [1 2 3 4 5]
Finished SMU Add operation

SUCCESS: install_add  Fri Oct 26 07:11:45 UTC 2035
Oct 26 07:11:46.695 %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
install add SMU flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
Device#
*Oct 26 07:11:46.656: %INSTALL-5-INSTALL_COMPLETED_INFO: Switch 1 R0/0: install_engine:
 Completed install add SMU flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
```

Verifying the additiong of the first SMU package file by using the **show install summary** command.

```
Device# show install summary
[ Switch 1 2 3 4 5 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
IMG   C    16.12.3.0.3752

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

Adding the second SMU package file.

```
Device# install add file flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin

install_add: START Fri Oct 26 07:12:38 UTC 2035
Oct 26 07:12:40.782 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 add flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
install_add: Adding SMU
install_add: Checking whether new add is allowed ....
```

```
--- Starting initial file syncing ---

*Oct 26 07:12:40.743: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
Started install add flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin[1]: Copying
flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin from switch 1 to switch 2 3 4 5
[2 3 4 5]: Finished copying to switch 2 switch 3 switch 4 switch 5
Info: Finished copying flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin to the
selected switch(es)
Finished initial file syncing

--- Starting SMU Add operation ---
Performing SMU_ADD on all members
  [1] SMU_ADD package(s) on switch 1
  [1] Finished SMU_ADD on switch 1
  [2] SMU_ADD package(s) on switch 2
  [2] Finished SMU_ADD on switch 2
  [3] SMU_ADD package(s) on switch 3
  [3] Finished SMU_ADD on switch 3
  [4] SMU_ADD package(s) on switch 4
  [4] Finished SMU_ADD on switch 4
  [5] SMU_ADD package(s) on switch 5
  [5] Finished SMU_ADD on switch 5
Checking status of SMU_ADD on [1 2 3 4 5]
SMU_ADD: Passed on [1 2 3 4 5]
Finished SMU Add operation

SUCCESS: install_add  Fri Oct 26 07:13:24 UTC 2035
Oct 26 07:13:25.656 %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
install add SMU flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
 Decive#
*Oct 26 07:13:25.616: %INSTALL-5-INSTALL_COMPLETED_INFO: Switch 1 R0/0: install_engine:
 Completed install add SMU flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
```

Verifying the addition and installation of both the SMU package files by using the **show install summary** command. The status of both package files is `I`, because they have not been activated and committed yet.

```
Device# show install summary

[ Switch 1 2 3 4 5 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
SMU   I    flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
IMG   C    16.12.3.0.3752

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

3. Activating the SMU package files.

   When entering multiple SMUs, use a comma (without a space before or after), to separate file names. Also ensure that total number of characters does not exceed 128. This step involves a reload.

```
Device# install activate file
flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin,cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin


install_activate: START Sun Oct 28 13:23:42 UTC 2035
```

```
Oct 28 13:23:44.620 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 activate
flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin,cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
install_activate: Activating SMU

*Oct 28 13:23:44.581: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
Started install activate
flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin,cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin

This operation may require a reload of the system. Do you want to proceed? [y/n]y
Executing pre scripts....

Executing pre sripts done.

--- Starting SMU Activate operation ---
Performing SMU_ACTIVATE on all members

*Oct 28 13:24:41.563: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 1 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 secondsOct 28 13:24:43.259:
 %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: R0/0: rollback_timer: Install auto abort
 timer will expire in 7200 seconds
*Oct 28 13:24:43.222: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 4 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds
*Oct 28 13:24:43.192: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 3 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds
*Oct 28 13:24:43.134: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 2 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds
*Oct 28 13:24:43.825: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 5 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds  [1] SMU_ACTIVATE
package(s) on switch 1
  [1] Finished SMU_ACTIVATE on switch 1
  [2] SMU_ACTIVATE package(s) on switch 2
  [2] Finished SMU_ACTIVATE on switch 2
  [3] SMU_ACTIVATE package(s) on switch 3
  [3] Finished SMU_ACTIVATE on switch 3
  [4] SMU_ACTIVATE package(s) on switch 4
  [4] Finished SMU_ACTIVATE on switch 4
  [5] SMU_ACTIVATE package(s) on switch 5
  [5] Finished SMU_ACTIVATE on switch 5
Checking status of SMU_ACTIVATE on [1 2 3 4 5]
SMU_ACTIVATE: Passed on [1 2 3 4 5]
Finished SMU Activate operation

install_activate: Reloading the box to complete activation of the SMU...
install_activate will reload the system now!

Chassis 4 reloading, reason - Reload command
reload fp action requested
rp processes exit with reload switch code



watchdog watchdog0: watchdog did not stop!
reboot: Restarting system

Initializing Hardware...

System Bootstrap, Version 16.12.1r [FC6], RELEASE SOFTWARE (P)
Compiled Thu 02/13/2020 12:36:08 by rel

Current ROMMON image : Primary
C9200L-24T-4G platform with 2097152 Kbytes of main memory

boot: attempting to boot from [flash:packages.conf]
```

```
boot: reading file packages.conf

##############
Oct 28 13:26:55.653: %BOOT-5-BOOTTIME_SMU_TEMP_ACTIVE_DETECTED: R0/0: install_engine:
SMU file /flash/cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin active temporary... SMU
 commit is pending
Oct 28 13:26:55.912: %BOOT-5-BOOTTIME_SMU_TEMP_ACTIVE_DETECTED: R0/0: install_engine:
SMU file /flash/cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin active temporary... SMU
 commit is pending

Waiting for 120 seconds for other switches to boot
##########
Switch number is 4
All switches in the stack have been discovered. Accelerating discovery
```

Verifying activation of the SMU package files by using the **show install summary** command. The status of both files is U, because they have not been committed yet.

```
Device# show install summary
[ Switch 1 2 3 4 5 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   U    flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
SMU   U    flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
IMG   C    16.12.3.0.3752

--------------------------------------------------------------------------------
Auto abort timer: active on install_activate, time before rollback - 01:50:16
--------------------------------------------------------------------------------
```

4. Committing the SMU package file

```
Device# install commit
install_commit: START Sun Oct 28 13:34:42 UTC 2035
Oct 28 13:34:45.202 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 commit

*Oct 28 13:34:45.146: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
Started install commitinstall_commit: Committing SMU
Executing pre scripts....
Executing pre sripts done.
--- Starting SMU Commit operation ---
Performing SMU_COMMIT on all members

*Oct 28 13:35:24.436: %PLATFORM-4-ELEMENT_WARNING: Switch 1 R0/0: smand: 5/RP/0: limited
 space - copy files out of flash: directory. flash: value 84% (1599 MB) exceeds warning
 level 70% (1337 MB).
*Oct 28 13:35:30.587: %PLATFORM-4-ELEMENT_WARNING: Switch 1 R0/0: smand: 2/RP/0: limited
 space - copy files out of flash: directory. flash: value 74% (1412 MB) exceeds warning
 level 70% (1337 MB).  [1] SMU_COMMIT package(s) on switch 1
  [1] Finished SMU_COMMIT on switch 1
  [2] SMU_COMMIT package(s) on switch 2
  [2] Finished SMU_COMMIT on switch 2
  [3] SMU_COMMIT package(s) on switch 3
  [3] Finished SMU_COMMIT on switch 3
  [4] SMU_COMMIT package(s) on switch 4
  [4] Finished SMU_COMMIT on switch 4
  [5] SMU_COMMIT package(s) on switch 5
  [5] Finished SMU_COMMIT on switch 5
Checking status of SMU_COMMIT on [1 2 3 4 5]
```

```
SMU_COMMIT: Passed on [1 2 3 4 5]
Finished SMU Commit operation

SUCCESS: install_commit /flash/cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
/flash/cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
Sun Oct 28 13:35:52 UTC 2035
Oct 28 13:35:53.789 %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
install commit SMU

JJ22-Vore_stack-24TE#
*Oct 28 13:35:53.749: %INSTALL-5-INSTALL_COMPLETED_INFO: Switch 1 R0/0: install_engine:
 Completed install commit SMU
```

Verifying the commit by using the **show install summary** command. The SMU package files have been installed, activated and committed, and the status is C.

```
Device# show install summary
[ Switch 1 2 3 4 5 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   C    flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
SMU   C    flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
IMG   C    16.12.3.0.3752

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

Example: Installing an SMU (3-Step Process, Using TFTP)

The following example shows how to install a SMU package by using the 3-step process. Here the SMU package file is saved in a remote (TFTP) location.

1. Adding the SMU package file.

```
Device# install add file
tftp://172.16.0.1//tftpboot/folder1/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

Jun 22 11:32:27.035: %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 add tftp://172.16.0.1//tftpboot/folder1/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Jun 22 11:32:27.035 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 add tftp://172.16.0.1//tftpboot/folder1/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Downloading file
tftp://172.16.0.1//tftpboot/folder1/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Finished downloading file
tftp://172.16.0.1//tftpboot/folder1/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin to
 flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
install_add: Adding SMU
install_add: Checking whether new add is allowed ....

--- Starting initial file syncing ---

025335: *Jun 22 2020 11:32:26 UTC: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0:
install_engine: Started install add
tftp://172.16.0.1//tftpboot/folder1/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin[1]:
 Copying flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin from switch 1 to switch
 2
[2]: Finished copying to switch 2
```

```
Info: Finished copying flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin to the
selected switch(es)
Finished initial file syncing

--- Starting SMU Add operation ---
Performing SMU_ADD on all members
[1] SMU_ADD package(s) on switch 1
[1] Finished SMU_ADD on switch 1
[2] SMU_ADD package(s) on switch 2
[2] Finished SMU_ADD on switch 2
Checking status of SMU_ADD on [1 2]
SMU_ADD: Passed on [1 2]
Finished SMU Add operation

SUCCESS: install_add Mon Jun 22 11:32:56 UTC 2020
Jun 22 11:32:57.598: %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
 install add SMU flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Jun 22 11:32:57.598 %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
install add SMU flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

ECSG-SEC-C9200-24P#
025336: *Jun 22 2020 11:32:57 UTC: %INSTALL-5-INSTALL_COMPLETED_INFO: Switch 1 R0/0:
install_engine: Completed install add SMU
flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
```

Verifying addition by using the **show install summary** command.

```
Device# show install summary
[ Switch 1 2 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type St Filename/Version
--------------------------------------------------------------------------------
SMU I flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
IMG C 16.12.02.0.6

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

2. Activating the SMU package file.

Note You use TFTP to add the SMU package file (in the previous step) and *flash*, to activate - not TFTP.

```
Device# install activate file flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

install_activate: START Mon Jun 22 11:37:17 UTC 2020

Jun 22 11:37:37.582: %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 activate flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Jun 22 11:37:37.582 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 activate flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
install_activate: Activating SMU

025337: *Jun 22 2020 11:37:37 UTC: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0:
install_engine: Started install activate
```

```
flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
This operation may require a reload of the system. Do you want to proceed? [y/n]n
```

Checking the version, by using the **show version** command:

```
Device# show version
Cisco IOS XE Software, Version 16.09.04
Cisco IOS Software [Fuji], Catalyst L3 Switch Software (CAT9K_LITE_IOSXE), Version 16.9.4,
 RELEASE SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2019 by Cisco Systems, Inc.
Compiled Thu 22-Aug-19 17:30 by mcpre
<output truncated>
```

3. Committing the SMU package file.

```
Device# install commit

install_commit: START Mon Jun 22 11:38:48 UTC 2020
SUCCESS: install_commit Mon Jun 22 11:38:52 UTC 2020
Device#
```

Verifying that the update package is now committed, and that it will be persistent across reloads:

```
Device# show install summary

Active Packages:
tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Inactive Packages:
No packages
Committed Packages:
tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Uncommitted Packages:
No packages
Device#
```

Example: Managing a SMU Package (Additional show commands, Rollback, Deactivation)

The following sample output displays information about active, inactive, committed, and uncommitted packages by using the **show install summary** command. Here SMU package file `cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin` is active and committed:

```
Device# show install summary

Active Packages:
    tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Inactive Packages:
   No packages
Committed Packages:
    tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Uncommitted Packages:
   No packages
Device#
```

The following is sample output from the **show install active** command:

```
Device# show install active

Active Packages:
```

```
tftp:cat3k-universalk9.2017-01-10_13.15.1.CSCxxx.SSA.dmp.bin
```

The following example shows how to rollback an update package to the committed package:

```
Device# install rollback to base

install_rollback: START Wed Jun 10 11:27:41 IST 2020
This rollback would require a reload. Do you want to proceed? [y/n]y
2 install_rollback: Reloading the box to take effect

Initializing Hardware ...
<after reload>
Device#
```

The following is sample output from the **show install summary** command:

```
Device# show install summary

Active Packages:
tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Inactive Packages:
No packages
Committed Packages:
tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Uncommitted Packages:
No packages
Device#
```

The following is sample output from the **show install log** command:

```
Device# show install log

[0|install_op_boot]: START Wed Jun 10 19:31:50 Universal 2020
[0|install_op_boot]: END SUCCESS Wed Jun 10 19:31:56 Universal 2020
```

The following example shows how to deactivate an SMU package file:

```
Device# install deactivate file tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

install_deactivate: START Wed Jun 10 10:49:07 IST 2020
The activation step would require a reload. Do you want to proceed? [y/n]y
Regular SMU. Reloading the box to complete activation of the SMU...

Initializing Hardware...
...
<after reload>
Device#
```

The following is sample output from the **show install summary** command:

```
Device# show install summary

Active Packages:
No packages
Inactive Packages:tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Committed Packages:
No packages
Uncommitted Packages:
No packages
```

```
Device#
```

The following example shows how to remove an SMU from the device:

```
Device# install remove file tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

install_remove: START Wed Jun 10 12:09:43 IST 2020
SUCCESS: install_remove /tftp/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin Wed Jun 10
12:09:49 IST 2020
Device#
```

The following is sample output from the **show install summary** command:

```
Device# show install summary

Active Packages:
No packages
Inactive Packages:
No packages
Committed Packages:
No packages
Uncommitted Packages:
No packages
```

Example: Managing an SMU

The following example shows how to copy an SMU file to TFTP:

```
Device# copy tftp://192.168.0.1//tftpboot/folder1/cat3k-
universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin

tftp:Destination filename [cat3k-
universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin]?

Accessing tftp://192.168.0.1//auto/tftpboot/folder1/cat3k-
universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin...
Loading /auto/tftpboot/folder1/cat3k-
universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin from
192.168.0.1 (via GigabitEthernet0): !
[OK - 17668 bytes]
17668 bytes copied in 0.058 secs (304621 bytes/sec)
```

The following is a sample output from the **show install summary** command:

```
Device# show install summary

[ Switch 1 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type St Filename/Version
--------------------------------------------------------------------------------
SMU C flash:cat3k-universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin
IMG C 16.6.3.0
```

The following example shows how to add a maintenance update package file:

```
Device# install add file tftp://192.168.0.1//tftpboot/folder1/cat3k-
universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin

install_add: START Sat Feb 26 14:06:04 PST 2017
SUCCESS: install_add tftp://192.168.0.1//tftpboot/folder1/cat3k-
universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin Sat Feb
26 14:06:12 PST 2017
Device#
```

The following is a sample output from the **show install summary** command after adding an SMU package file to the device:

```
Device# show install summary

Active Packages:
No packages
Inactive Packages:
tftp:cat3k-universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin
Committed Packages:
No packages
Uncommitted Packages:
No packages
Device#
```

The following example shows how to activate an added SMU package file:

```
Device# install activate file tftp://192.168.0.1//tftpboot/folder1/cat3k-
universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin

install_activate: START Sat Feb 26 14:10:55 PST 2017
The activation step would require a reload. Do you want to proceed? [y/n]y
Regular SMU. Reloading the box to complete activation of the SMU...
Feb 26 14:11:23.873 R0/0: %PMAN-5-EXITACTION: Process manager is exiting:
reload action requested
Initializing Hardware ...
Checking for PCIe device presence...done
System integrity status: 0x610
Rom image verified correctly
<after reload>
Device#
```

The following is a sample output from the **show version** command:

```
Device# show version

Cisco IOS XE Software, Version BLD_POLARIS_DEV_SMU_LATEST_20170110_13.15.1 -
SMU-PATCHED
Cisco IOS Software [Everest], Catalyst L3 Switch Software (CAT3K_CAA-UNIVERSALK9-M),
Experimental Version 16.6.20170110_13.15.1 [BLD_V166_SMU_LATEST_20170127_13.15.1 SMU-PATCHED]
Copyright (c) 1986-2017 by Cisco Systems, Inc.
Compiled Sat 26-Feb-17 16:07 by mcpre
...
```

The following is a sample output from the **show install summary** command displays the status of the model package as active and uncommitted:

```
Device# show install summary

Active Packages:
    tftp:cat3k-universalk9.2017-01-10_13.15.1.CSCxxx.SSA.dmp.bin
Inactive Packages:
```

```
   No packages
Committed Packages:
    tftp:cat3k-universalk9.2017-01-10_13.15.1.CSCxxx.SSA.dmp.bin
Uncommitted Packages:
   No packages
Device#
```

The following is a sample output from the **show install active** command:

```
Device# show install active

Active Packages:
tftp:cat3k-universalk9.2017-01-10_13.15.1.CSCxxx.SSA.dmp.bin
```

The following example shows how to execute the **install commit** command:

```
Device# install commit

install_commit: START Sat Feb 26 06:46:48 UTC 2017
SUCCESS: install_commit Sat Feb 26 06:46:52 UTC 2017
Device#
```

The following is a sample output from the **show install summary** command displays that the update package is now committed, and that it will be persistent across reloads:

```
Device# show install summary

Active Packages:
tftp:cat3k-universalk9.2017-01-10_13.15.1.CSCxxx.SSA.dmp.bin
Inactive Packages:
No packages
Committed Packages:
tftp:cat3k-universalk9.2017-01-10_13.15.1.CSCxxx.SSA.dmp.bin
Uncommitted Packages:
No packages
Device#
```

The following example shows how to rollback an update package to the committed package:

```
Device# install rollback to base

install_rollback: START Sat Feb 26 11:27:41 PST 2017
This rollback would require a reload. Do you want to proceed? [y/n]y
2 install_rollback: Reloading the box to take effect

Initializing Hardware ...
<after reload>
Device#
```

The following is a sample output from the **show install summary** command:

```
Device# show install summary

Active Packages:
tftp:cat3k-universalk9.2017-01-10_13.15.1.CSCxxx.SSA.dmp.bin
Inactive Packages:
No packages
Committed Packages:
tftp:cat3k-universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin
Uncommitted Packages:
No packages
```

```
Device#
```

The following is a sample output from the **show install log** command:

```
Device# show install log

[0|install_op_boot]: START Sat Feb 26 19:31:50 Universal 2017
[0|install_op_boot]: END SUCCESS Sat Feb 26 19:31:56 Universal 2017
```

The following example shows how to deactivate an SMU package file:

```
Device# install deactivate file tftp:cat3k-
universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin

install_deactivate: START Sat Feb 26 10:49:07 PST 2017
The activation step would require a reload. Do you want to proceed? [y/n]y
Regular SMU. Reloading the box to complete activation of the SMU...

Initializing Hardware...
...
<after reload>
Device#
```

The following is a sample output from the **show install summary** command:

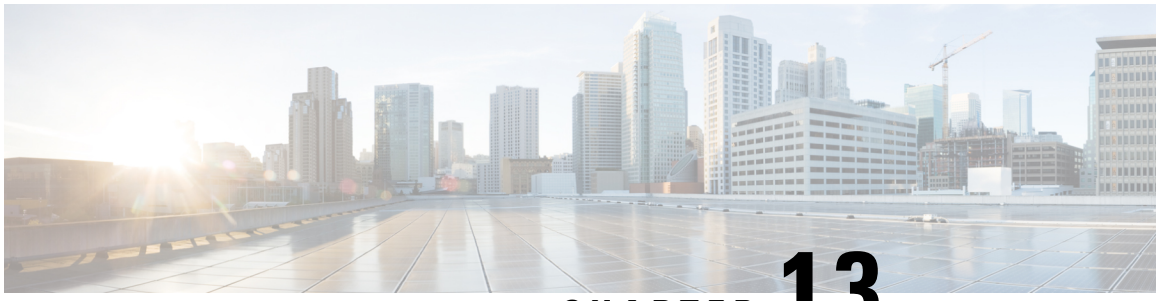
```
Device# show install summary

Active Packages:
No packages
Inactive Packages:tftp:cat3k-universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin
Committed Packages:
No packages
Uncommitted Packages:
No packages
Device#
```

The following example shows how to remove an SMU from the device:

```
Device# install remove file tftp:cat3k-
universalk9.2017-01-10_13.15.1.CSCxxxxxxx.SSA.dmp.bin

install_remove: START Sat Feb 26 12:09:43 PST 2017
SUCCESS: install_remove /tftp/cat3k-universalk9.2017-01-10_13.15.1.
CSCxxxxxxx.SSA.dmp.bin Sat Feb 26 12:09:49 PST 2017
Device#
```

The following is a sample output from the **show install summary** command:

```
Device# show install summary

Active Packages:
No packages
Inactive Packages:
No packages
Committed Packages:
No packages
Uncommitted Packages:
No packages
```



CHAPTER 13

Working with the Flash File System

Information About the Flash File System

The flash file system is a single flash device on which you can store files. It also provides several commands to help you manage software bundles and configuration files. The default flash file system on the device is named flash:.

As viewed from the active device, flash: refers to the local flash device, which is the device attached to the same device on which the file system is being viewed.

Only one user at a time can manage the software bundles and configuration files.

Displaying Available File Systems

To display the available file systems on your device, use the **show file systems** privileged EXEC command as shown in this example for a standalone device:

```
Device# show file systems
Size(b) Free(b) Type Flags Prefixes
- - opaque rw system:
- - opaque rw tmpsys:
1651314688 1559785472 disk rw crashinfo:
* 11353194496 9693396992 disk rw flash:
8049967104 7959392256 disk ro webui:
- - opaque rw null:
- - opaque ro tar:
- - network rw tftp:
2097152 2080848 nvram rw nvram:
- - opaque wo syslog:
- - network rw rcp:
```

```
- - network rw http:
- - network rw ftp:
- - network rw scp:
- - network rw https:
- - opaque ro cns:
```

This example displays the usbflash1 filesystem format.

```
Device#show usbflash1: filesys
Filesystem: usbflash1
Filesystem Path: /vol/usb1
Filesystem Type: ext4
Mounted: Read/Write
```

Table 14: show file systems Field Descriptions

Field	Value
Size(b)	Amount of memory in the file system in bytes.
Free(b)	Amount of free memory in the file system in bytes.
Type	Type of file system. **disk**—The file system is for a flash memory device, USB flash, and crashinfo file. **network**—The file system for network devices; for example, an FTP server or and HTTP server. **nvram**—The file system is for a NVRAM device. **opaque**—The file system is a locally generated pseudo file system (for example, the system) or a download interface, such as brimux. **unknown**—The file system is an unknown type.
Flags	Permission for file system. **ro**—read-only. **rw**—read/write. **wo**—write-only.

Field	Value
Prefixes	Alias for file system. **crashinfo:**—Crashinfo file. **flash:**—Flash file system. **ftp:**—FTP server. **http:**—HTTP server. **https:**—Secure HTTP server. **nvram:**—NVRAM. **null:**—Null destination for copies. You can copy a remote file to null to find its size. **rcp:**—Remote Copy Protocol (RCP) server. **scp:**—Session Control Protocol (SCP) server. **system:**—Contains the system memory, including the running configuration. **tftp:**—TFTP network server. **usbflash0:**—USB flash memory. **usbflash1:**—External USB flash memory. **ymodem:**—Obtain the file from a network machine by using the Ymodem protocol.

Setting the Default File System

You can specify the file system or directory that the system uses as the default file system by using the **cd** *filesystem:* privileged EXEC command. You can set the default file system to omit the *filesystem:* argument from related commands. For example, for all privileged EXEC commands that have the optional *filesystem:* argument, the system uses the file system specified by the **cd** command.

By default, the default file system is *flash:*.

You can display the current default file system as specified by the **cd** command by using the **pwd** privileged EXEC command.

Displaying Information About Files on a File System

You can view a list of the contents of a file system before manipulating its contents. For example, before copying a new configuration file to flash memory, you might want to verify that the file system does not already contain a configuration file with the same name. Similarly, before copying a flash configuration file to another location, you might want to verify its filename for use in another command. To display information about files on a file system, use one of the privileged EXEC commands listed in the following table.

Table 15: Commands for Displaying Information About Files

Command	Description
dir [/all] **[***filesystem:filename***]**	Displays a list of files on a file system.
show file systems	Displays more information about each of the files on a file system.
show file information *file-url*	Displays information about a specific file.
show file descriptors	Displays a list of open file descriptors. File descriptors are the internal representations of open files. You can use this command to see if another user has a file open.

For example, to display a list of all files in a file system, use the **dir** privileged EXEC command:

```
Device# dir flash:
DDirectory of bootflash:/

616513  drwx             4096  Jul 15 2015 07:11:35 +00:00  .installer
608402  -rw-            33818  Sep 25 2015 11:41:35 +00:00  bootloader_evt_handle.log
608403  drwx             4096  Feb 27 2017 13:56:47 +00:00  .ssh
608410  -rw-                0   Jun 5 2015 10:16:17 +00:00  dc_stats.txt
608411  drwx            20480  Sep 23 2015 11:50:13 +00:00  core
624625  drwx             4096  Sep 23 2015 12:29:27 +00:00  .prst_sync
640849  drwx             4096  Feb 27 2017 13:57:30 +00:00  .rollback_timer
608412  drwx             4096  Jun 17 2015 18:12:47 +00:00  orch_test_logs
608413  -rw-         33554432  Sep 25 2015 11:43:15 +00:00  nvram_config
608417  -rw-               35  Sep 25 2015 20:17:42 +00:00  pnp-tech-time
608439  -rw-           214054  Sep 25 2015 20:17:48 +00:00  pnp-tech-discovery-summary
608419  drwx             4096  Jul 23 2015 07:50:25 +00:00  util
616514  drwx             4096  Mar 18 2015 11:09:04 +00:00  onep
608442  -rw-              556  Mar 18 2015 11:19:34 +00:00  vlan.dat
608448  -rw-          1131779  Mar 28 2015 13:13:48 +00:00  log.txt
616516  drwx             4096   Apr 1 2015 09:34:56 +00:00  gs_script
616517  drwx             4096   Apr 6 2015 09:42:38 +00:00  tools
608440  -rw-              252  Sep 25 2015 11:41:52 +00:00  boothelper.log
624626  drwx             4096  Apr 17 2015 06:10:55 +00:00  SD_AVC_AUTO_CONFIG
608488  -rw-            98869  Sep 25 2015 11:42:15 +00:00  memleak.tcl
608437  -rwx            17866  Jul 16 2015 04:01:10 +00:00  ardbeg_x86
632745  drwx             4096  Aug 20 2015 11:35:09 +00:00  CRDU
632746  drwx             4096  Sep 16 2015 08:57:44 +00:00  ardmore
608418  -rw-          1595361   Jul 8 2015 11:18:33 +00:00
system-report_RP_0_20150708-111832-UTC.tar.gz
608491  -rw-         67587176  Aug 12 2015 05:30:35 +00:00  mcln_x86_kernel_20170628.SSA
608492  -rwx         74880100  Aug 12 2015 05:30:57 +00:00  stardust.x86.idprom.0718B

11250098176 bytes total (9128050688 bytes free)
Device#
```

Changing Directories and Displaying the Working Directory

Follow these steps to change directories and to display the working directory:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**dir** *filesystem:* **Example:** `Device# dir flash:`	Displays the directories on the specified file system. For *filesystem:*, use flash: for the system board flash device.
Step 3	**cd** *directory_name* **Example:** `Device# cd new_configs`	Navigates to the specified directory. The command example shows how to navigate to the directory named *new_configs*.
Step 4	**pwd** **Example:** `Device# pwd`	Displays the working directory.
Step 5	**cd** **Example:** `Device# cd`	Navigates to the default directory.

Creating Directories

Beginning in privileged EXEC mode, follow these steps to create a directory:

Procedure

	Command or Action	Purpose
Step 1	**dir** *filesystem:* **Example:** `Device# dir flash:`	Displays the directories on the specified file system. For *filesystem:*, use flash: for the system board flash device.
Step 2	**mkdir** *directory_name* **Example:** `Device# mkdir new_configs`	Creates a new directory. Directory names are case sensitive and are limited to 45 characters between the slashes (/); the name cannot contain control characters, spaces, slashes, quotes, semicolons, or colons.

	Command or Action	Purpose
Step 3	**dir** *filesystem:* **Example:** `Device# dir flash:`	Verifies your entry.

Removing Directories

To remove a directory with all its files and subdirectories, use the **delete /force /recursive** *filesystem:/file-url* privileged EXEC command.

Use the **/recursive** keyword to delete the named directory and all subdirectories and the files contained in it. Use the **/force** keyword to suppress the prompting that confirms a deletion of each file in the directory. You are prompted only once at the beginning of this deletion process.

For *filesystem*, use **flash:** for the system board flash device. For *file-url*, enter the name of the directory to be deleted. All of the files in the directory and the directory are removed.

Caution When directories are deleted, their contents cannot be recovered.

Copying Files

To copy a file from a source to a destination, use the **copy** *source-url destination-url* privileged EXEC command. For the source and destination URLs, you can use **running-config** and **startup-config** keyword shortcuts. For example, the **copy running-config startup-config** command saves the currently running configuration file to the NVRAM section of flash memory to be used as the configuration during system initialization.

Network file system URLs include ftp:, rcp:, tftp:, scp:, http:, and https: and have these syntaxes:

- ftp:[[//username [:password]@location]/directory]/filename
- rcp:[[//username@location]/directory]/filename
- tftp:[[//location]/directory]/filename
- scp:[[//username [:password]@location]/directory]/filename
- http:[[//username [:password]@location]/directory]/filename
- https:[[//username [:password]@location]/directory]/filename

Note The password must not contain the special character '@'. If the character '@' is used, the copy fails to parse the IP address of the server.

Local writable file systems include flash:.

Some invalid combinations of source and destination exist. Specifically, you cannot copy these combinations:

- From a running configuration to a running configuration
- From a startup configuration to a startup configuration

Deleting Files

When you no longer need a file on a flash memory device, you can permanently delete it. To delete a file or directory from a specified flash device, use the **delete** [**/force**] [**/recursive**] [*filesystem:*]*/file-url* privileged EXEC command.

Use the **/recursive** keyword for deleting a directory and all subdirectories and the files contained in it. Use the **/force** keyword to suppress the prompting that confirms a deletion of each file in the directory. You are prompted only once at the beginning of this deletion process. Use the **/force** and **/recursive** keywords for deleting old software images that were installed by using the **archive download-sw** command but are no longer needed.

If you omit the *filesystem:* option, the device uses the default device specified by the **cd** command. For *file-url*, you specify the path (directory) and the name of the file to be deleted.

When you attempt to delete any files, the system prompts you to confirm the deletion.

Caution When files are deleted, their contents cannot be recovered.

This example shows how to delete the file *myconfig* from the default flash memory device:

```
Device# delete myconfig
```

Creating, Displaying and Extracting Files

You can create a file and write files into it, list the files in a file, and extract the files from a file as described in the next sections.

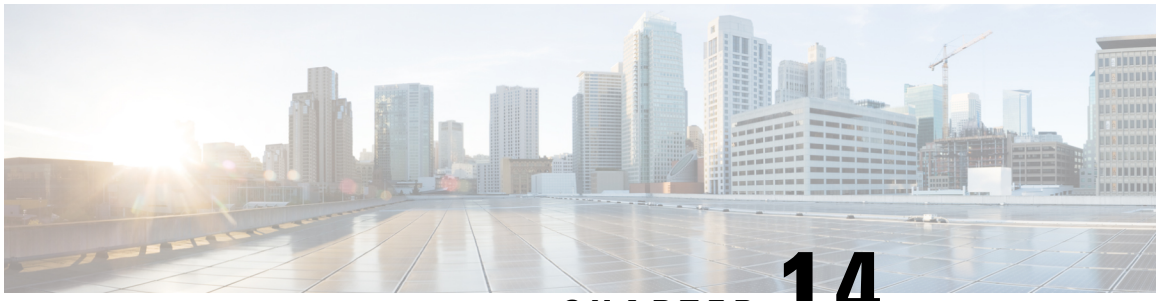
Beginning in privileged EXEC mode, follow these steps to create a file, display the contents, and extract it:

Procedure

	Command or Action	Purpose
Step 1	**archive tar /create** *destination-url* **flash:** */file-url* **Example:** `Device# archive tar /create tftp:192.168.10.30/saved. flash:/new-configs`	Creates a file and adds files to it. For destination-url, specify the destination URL alias for the local or network file system and the name of the file to create: • Local flash file system syntax: **flash:** • FTP syntax: **ftp**:[[//*username*[:*password*]@*location*]/*directory*]/*-filename.*

	Command or Action	Purpose
		• RCP syntax: **rcp**:[[//*username@location*]/*directory*]/*-filename*. • TFTP syntax: **tftp**:[[//*location*]/*directory*]/*-filename*. For **flash:***/file-url*, specify the location on the local flash file system in which the new file is created. You can also specify an optional list of files or directories within the source directory to add to the new file. If none are specified, all files and directories at this level are written to the newly created file.
Step 2	**archive tar /table** *source-url* **Example:** `Device# archive tar /table flash: /new_configs`	Displays the contents of a file. For *source-url*, specify the source URL alias for the local or network file system. The *-filename.* is the file to display. These options are supported: • Local flash file system syntax: **flash:** • FTP syntax: **ftp**:[[//*username*[:*password*]@*location*]/*directory*]/*-filename.* • RCP syntax: **rcp**:[[//*username@location*]/*directory*]/*-filename*. • TFTP syntax: **tftp**:[[//*location*]/*directory*]/*-filename*. You can also limit the file displays by specifying a list of files or directories after the file. Only those files appear. If none are specified, all files and directories appear.
Step 3	**archive tar /xtract** *source-url* **flash:**/*file-url* [*dir/file...*] **Example:** `Device# archive tar /xtract tftp:/192.168.10.30/saved. flash:/new-configs`	Extracts a file into a directory on the flash file system. For *source-url*, specify the source URL alias for the local file system. The *-filename.* is the file from which to extract files. These options are supported: • Local flash file system syntax: **flash:** • FTP syntax: **ftp**:[[//*username*[:*password*]@*location*]/*directory*]/*-filename.* • RCP syntax: **rcp**:[[//*username@location*]/*directory*]/*-filename*. • TFTP syntax: **tftp**:[[//*location*]/*directory*]/*-filename*. For **flash:**/*file-url* [*dir/file...*], specify the location on the local flash file system from which the file is extracted. Use

	Command or Action	Purpose
		the *dir/file...* option to specify a list of files or directories within the file to be extracted. If none are specified, all files and directories are extracted.
Step 4	**more** [/**ascii** | /**binary** | /**ebcdic**] /*file-url* **Example:** `Device# more flash:/new-configs`	Displays the contents of any readable file, including a file on a remote file system.



CHAPTER 14

Configuring Secure Storage

Information About Secure Storage

Secure Storage feature allows you to secure critical configuration information by encrypting it. It encrypts asymmetric key-pairs, pre-shared secrets, the type 6 password encryption key and certain credentials. An instance-unique encryption key is stored in the hardware trust anchor to prevent it from being compromised.

Enabling Secure Storage

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters the global configuration mode.
Step 2	**service private-config-encryption** **Example:** `DEvice(config)# service private-config-encryption`	Enables the Secure Storage feature on your device.
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 4	**write memory** **Example:**	Encrypts the private-config file and saves the file in an encrypted format.

	Command or Action	Purpose
	`Device# write memory`	

Disabling Secure Storage

Before you begin

To disable Secure Storage feature on a device, perform this task:

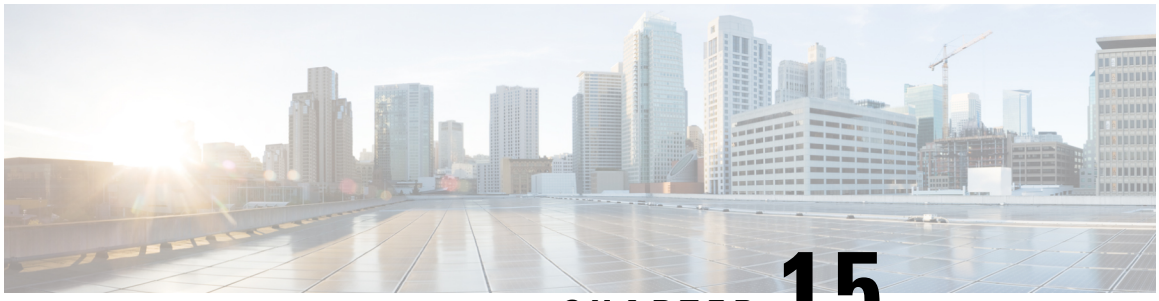
Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters the global configuration mode.
Step 2	**no service private-config-encryption** **Example:** `Device(config)# no service private-config-encryption`	Disables the Secure Storage feature on your device. When secure storage is disabled, all the user data is stored in plain text in the NVRAM.
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 4	**write memory** **Example:** `Device# write memory`	Decrypts the private-config file and saves the file in plane format.

Verifying the Status of Encryption

Use the **show parser encrypt file status** command to verify the status of encryption. The following command output indicates that the feature is available but the file is not encrypted. The file is in 'plain text' format.

```
Device#show parser encrypt file status
Feature: Enabled
File Format: Plain Text
Encryption Version: Ver1
```



CHAPTER 15

Troubleshooting the Software Configuration

This chapter describes how to identify and resolve software problems related to the Cisco IOS software on the switch. Depending on the nature of the problem, you can use the command-line interface (CLI), Device Manager, or Network Assistant to identify and solve problems.

Additional troubleshooting information, such as LED descriptions, is provided in the hardware installation guide.

Information About Troubleshooting the Software Configuration

Software Failure on a Switch

Switch software can be corrupted during an upgrade by downloading the incorrect file to the switch, and by deleting the image file. In all of these cases, there is no connectivity.Follow the steps described in the Recovering from a Software Failure, on page 220 section to recover from a software failure.

Lost or Forgotten Password on a Device

The default configuration for the device allows an end user with physical access to the device to recover from a lost password by interrupting the boot process during power-on and by entering a new password. These recovery procedures require that you have physical access to the device.

Note On these devices, a system administrator can disable some of the functionality of this feature by allowing an end user to reset a password only by agreeing to return to the default configuration. If you are an end user trying to reset a password when password recovery has been disabled, a status message reminds you to return to the default configuration during the recovery process.

Note You cannot recover encryption password key, when Cisco WLC configuration is copied from one Cisco WLC to another (in case of an RMA).

Ping

The device supports IP ping, which you can use to test connectivity to remote hosts. Ping sends an echo request packet to an address and waits for a reply. Ping returns one of these responses:

- Normal response—The normal response (*hostname* is alive) occurs in 1 to 10 seconds, depending on network traffic.
- Destination does not respond—If the host does not respond, a *no-answer* message is returned.
- Unknown host—If the host does not exist, an *unknown host* message is returned.
- Destination unreachable—If the default gateway cannot reach the specified network, a *destination-unreachable* message is returned.
- Network or host unreachable—If there is no entry in the route table for the host or network, a *network or host unreachable* message is returned.

Refere the section Executing Ping, on page 222 to understand how **ping** works.

Layer 2 Traceroute

The Layer 2 traceroute feature allows the switch to identify the physical path that a packet takes from a source device to a destination device. Layer 2 traceroute supports only unicast source and destination MAC addresses. Traceroute finds the path by using the MAC address tables of the devices in the path. When the Device detects a device in the path that does not support Layer 2 traceroute, the Device continues to send Layer 2 trace queries and lets them time out.

The Device can only identify the path from the source device to the destination device. It cannot identify the path that a packet takes from source host to the source device or from the destination device to the destination host.

Layer 2 Traceroute Guidelines

- Cisco Discovery Protocol (CDP) must be enabled on all the devices in the network. For Layer 2 traceroute to function properly, do not disable CDP.

  If any devices in the physical path are transparent to CDP, the switch cannot identify the path through these devices.
- A device is reachable from another device when you can test connectivity by using the **ping** privileged EXEC command. All devices in the physical path must be reachable from each other.
- The maximum number of hops identified in the path is ten.
- You can enter the **traceroute mac** or the **traceroute mac ip** privileged EXEC command on a device that is not in the physical path from the source device to the destination device. All devices in the path must be reachable from this switch.

- The **traceroute mac** command output shows the Layer 2 path only when the specified source and destination MAC addresses belong to the same VLAN. If you specify source and destination MAC addresses that belong to different VLANs, the Layer 2 path is not identified, and an error message appears.
- If you specify a multicast source or destination MAC address, the path is not identified, and an error message appears.
- If the source or destination MAC address belongs to multiple VLANs, you must specify the VLAN to which both the source and destination MAC addresses belong. If the VLAN is not specified, the path is not identified, and an error message appears.
- The **traceroute mac ip** command output shows the Layer 2 path when the specified source and destination IP addresses belong to the same subnet. When you specify the IP addresses, the device uses the Address Resolution Protocol (ARP) to associate the IP addresses with the corresponding MAC addresses and the VLAN IDs.
  - If an ARP entry exists for the specified IP address, the device uses the associated MAC address and identifies the physical path.
  - If an ARP entry does not exist, the device sends an ARP query and tries to resolve the IP address. If the IP address is not resolved, the path is not identified, and an error message appears.
- When multiple devices are attached to one port through hubs (for example, multiple CDP neighbors are detected on a port), the Layer 2 traceroute feature is not supported. When more than one CDP neighbor is detected on a port, the Layer 2 path is not identified, and an error message appears.
- This feature is not supported in Token Ring VLANs.
- Layer 2 traceroute opens a listening socket on the User Datagram Protocol (UDP) port 2228 that can be accessed remotely with any IPv4 address, and does not require any authentication. This UDP socket allows to read VLAN information, links, presence of particular MAC addresses, and CDP neighbor information, from the device. This information can be used to eventually build a complete picture of the Layer 2 network topology.
- Layer 2 traceroute is enabled by default and can be disabled by running the **no l2 traceroute** command in global configuration mode. To re-enable Layer 2 traceroute, use the **l2 traceroute** command in global configuration mode.

IP Traceroute

You can use IP traceroute to identify the path that packets take through the network on a hop-by-hop basis. The command output displays all network layer (Layer 3) devices, such as routers, that the traffic passes through on the way to the destination.

Your Device can participate as the source or destination of the **traceroute** privileged EXEC command and might or might not appear as a hop in the **traceroute** command output. If the Device is the destination of the traceroute, it is displayed as the final destination in the traceroute output. Intermediate devices do not show up in the traceroute output if they are only bridging the packet from one port to another within the same VLAN. However, if the intermediate Device is a multilayer Device that is routing a particular packet, this device shows up as a hop in the traceroute output.

The **traceroute** privileged EXEC command uses the Time To Live (TTL) field in the IP header to cause routers and servers to generate specific return messages. Traceroute starts by sending a User Datagram Protocol (UDP) datagram to the destination host with the TTL field set to 1. If a router finds a TTL value of 1 or 0, it

drops the datagram and sends an Internet Control Message Protocol (ICMP) time-to-live-exceeded message to the sender. Traceroute finds the address of the first hop by examining the source address field of the ICMP time-to-live-exceeded message.

To identify the next hop, traceroute sends a UDP packet with a TTL value of 2. The first router decrements the TTL field by 1 and sends the datagram to the next router. The second router sees a TTL value of 1, discards the datagram, and returns the time-to-live-exceeded message to the source. This process continues until the TTL is incremented to a value large enough for the datagram to reach the destination host (or until the maximum TTL is reached).

To learn when a datagram reaches its destination, traceroute sets the UDP destination port number in the datagram to a very large value that the destination host is unlikely to be using. When a host receives a datagram destined to itself containing a destination port number that is unused locally, it sends an ICMP *port-unreachable* error to the source. Because all errors except port-unreachable errors come from intermediate hops, the receipt of a port-unreachable error means that this message was sent by the destination port.

Go to Example: Performing a Traceroute to an IP Host, on page 227 to see an example of IP traceroute process.

Time Domain Reflector Guidelines

You can use the Time Domain Reflector (TDR) feature to diagnose and resolve cabling problems. When running TDR, a local device sends a signal through a cable and compares the reflected signal to the initial signal.

TDR can detect these cabling problems:

- Open, broken, or cut twisted-pair wires—The wires are not connected to the wires from the remote device.
- Shorted twisted-pair wires—The wires are touching each other or the wires from the remote device. For example, a shorted twisted pair can occur if one wire of the twisted pair is soldered to the other wire.

  If one of the twisted-pair wires is open, TDR can find the length at which the wire is open.

Use TDR to diagnose and resolve cabling problems in these situations:

- Replacing a device.
- Setting up a wiring closet
- Troubleshooting a connection between two devices when a link cannot be established or when it is not operating properly

When you run TDR, the device reports accurate information in these situations:

- The cable for the gigabit link is a solid-core cable.
- The open-ended cable is not terminated.

When you run TDR, the device does not report accurate information in these situations:

- The cable for the gigabit link is a twisted-pair cable or is in series with a solid-core cable.
- The link is a 10-megabit or a 100-megabit link.
- The cable is a stranded cable.
- The link partner is a Cisco IP Phone.

- The link partner is not IEEE 802.3 compliant.

Go to Running TDR and Displaying the Results, on page 223 to know the TDR commands.

Debug Commands

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. It is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

All **debug** commands are entered in privileged EXEC mode, and most **debug** commands take no arguments.

System Report

System reports or crashinfo files save information that helps Cisco technical support representatives to debug problems that caused the Cisco IOS image to fail (crash). It is necessary to quickly and reliably collect critical crash information with high fidelity and integrity. Further, it is necessary to collect this information and bundle it in a way that it can be associated or identified with a specific crash occurrence.

The system does not generate reports in case of a reload.

During a process crash, the following is collected locally from the switch:

1. Full process core
2. Tracelogs
3. IOS syslogs (not guaranteed in case of non-active crashes)
4. System process information
5. Bootup logs
6. Reload logs
7. Certain types of /proc information

This information is stored in separate files which are then archived and compressed into one bundle. This makes it convenient to get a crash snapshot in one place, and can be then moved off the box for analysis. This report is generated before the switch goes down to rommon/bootloader.

Except for the full core and tracelogs, everything else is a text file.

Use the **request platform software process core fed active** command to generate the core dump.

```
Switch# request platform software process core fed active
Process : fed main event (28155) encountered fatal signal 6
Process : fed main event stack :

SUCCESS: Core file generated.

h2-macallan1#dir bootflash:core
Directory of bootflash:/core/
```

```
178483  -rw-               1  May 23 2017 06:05:17 +00:00  .callhome
194710  drwx            4096  Aug 16 2017 19:42:33 +00:00  modules
178494  -rw-         10829893  Aug 23 2017 09:46:23 +00:00
switch_RP_0_fed_28155_20170823-094616-UTC.core.gz
```

Crashinfo Files

By default the system report file will be generated and saved into the /crashinfo directory. Ifit cannot be saved to the crashinfo partition for lack of space, then it will be saved to the /flash directory.

To display the files, enter the **dir crashinfo:** command. The following is sample output of a crashinfo directory:

System reports are located in the crashinfo directory in the following format:

```
system-report_[switch number]_[date]-[timestamp]-UTC.gz
```

After a switch crashes, check for a system report file. The name of the most recently generated system report file is stored in the last_systemreport file under the crashinfo directory. The system report and crashinfo files assist TAC while troubleshooting the issue.

The system report generated can be further copied using TFTP, HTTP and few other options.

```
Switch#copy crashinfo: ?
crashinfo:     Copy to crashinfo: file system
flash:         Copy to flash: file system
ftp:           Copy to ftp: file system
http:          Copy to http: file system
https:         Copy to https: file system
null:          Copy to null: file system
nvram:         Copy to nvram: file system
rcp:           Copy to rcp: file system
running-config Update (merge with) current system configuration
scp:           Copy to scp: file system
startup-config Copy to startup configuration
syslog:        Copy to syslog: file system
system:        Copy to system: file system
tftp:          Copy to tftp: file system
tmpsys:        Copy to tmpsys: file system
```

The general syntax for copying onto TFTP server is as follows:

```
Switch#copy crashinfo: tftp:
Source filename [system-report_1_20150909-092728-UTC.gz]?
Address or name of remote host []? 1.1.1.1
Destination filename [system-report_1_20150909-092728-UTC.gz]?
```

The tracelogs can be collected by issuing a trace archive command. This command provides time period options. The command syntax is as follows:

```
Switch#request platform software trace archive ?
last     Archive trace files of last x days
target   Location and name for the archive file
```

The tracelogs stored in crashinfo: or flash: directory from within the last 3650 days can be collected.

```
Switch# request platform software trace archive last ?
<1-3650> Number of days (1-3650)
Switch#request platform software trace archive last 3650 days target ?
crashinfo: Archive file name and location
flash:     Archive file name and location
```

Note It is important to clear the system reports or trace archives from flash or crashinfo directory once they are copied out, in order to have space available for tracelogs and other purposes.

In a complex network it is difficult to track the origin of a system-report file. This task is made easier if the system-report files are uniquely identifiable. The hostname will be prepended to the system-report file name making the reports uniquely identifiable.

The following example displays system-report files with the hostname prepended:

```
Switch#dir flash:/core | grep HOSTNAME
40486  -rw-        108268293  Oct 21 2019 16:07:50 -04:00
HOSTNAME-system-report_20191021-200748-UTC.tar.gz
40487  -rw-            17523  Oct 21 2019 16:07:56 -04:00
HOSTNAME-system-report_20191021-200748-UTC-info.txt
40484  -rw-         48360998  Oct 21 2019 16:55:24 -04:00
HOSTNAME-system-report_20191021-205523-UTC.tar.gz
40488  -rw-            14073  Oct 21 2019 16:55:26 -04:00
HOSTNAME-system-report_20191021-205523-UTC-info.txt
```

Onboard Failure Logging on the Switch

You can use the onboard failure logging (OBFL) feature to collect information about the device. The information includes uptime, temperature, and voltage information and helps Cisco technical support representatives to troubleshoot device problems. We recommend that you keep OBFL enabled and do not erase the data stored in the flash memory.

By default, OBFL is enabled. It collects information about the device and small form-factor pluggable (SFP) modules. The device stores this information in the flash memory:

- CLI commands—Record of the OBFL CLI commands that are entered on a standalone device.
- Message—Record of the hardware-related system messages generated by a standalone device .
- Temperature—Temperature of a standalone deicev .
- Uptime data—Time when a standalone device starts, the reason the device restarts, and the length of time the device has been running since it last restarted.
- Voltage—System voltages of a standalone device .

You should manually set the system clock or configure it by using Network Time Protocol (NTP).

When the device is running, you can retrieve the OBFL data by using the **show logging onboard** privileged EXEC commands. If the device fails, contact your Cisco technical support representative to find out how to retrieve the data.

When an OBFL-enabled device is restarted, there is a 10-minute delay before logging of new data begins.

Possible Symptoms of High CPU Utilization

Excessive CPU utilization might result in these symptoms, but the symptoms might also result from other causes, some of which are the following:

- Spanning tree topology changes

- EtherChannel links brought down due to loss of communication
- Failure to respond to management requests (ICMP ping, SNMP timeouts, slow Telnet or SSH sessions)
- UDLD flapping
- IP SLAs failures because of SLAs responses beyond an acceptable threshold
- DHCP or IEEE 802.1x failures if the switch does not forward or respond to requests

How to Troubleshoot the Software Configuration

Recovering from a Software Failure

Before you begin

This recovery procedure requires that you have physical access to the switch.

This procedure uses boot loader commands and TFTP to recover from a corrupted or incorrect image file.

Step 1 From your PC, download the software image file (*image.bin*) from Cisco.com.

Step 2 Load the software image to your TFTP server.

Step 3 Connect your PC to the switch Ethernet management port.

Step 4 Power down the switch.

Step 5 Reconnect the power and press Ctrl-C when the device is preparing to autoboot. This brings the device to rommon mode.

Example:

```
Last reset cause: SoftwareResetTrig
ESS9300 platform with 16777216 Kbytes of main memory

Preparing to autoboot. [Press Ctrl-C to interrupt]  3    (interrupted)
switch:
switch:
```

Step 6 From the bootloader prompt, ensure that you can ping your TFTP server.

a) Set switch IP address: **set IP_ADDRESS** *ip_address*

Example:

```
switch: set IP_ADDRESS 192.168.2.123
```

b) Set switch subnet mask: **set IP_SUBNET_MASK** *subnet_mask*

Example:

```
switch: set IP_SUBNET_MASK 255.255.255.0
```

c) Set default gateway: **set DEFAULT_GATEWAY** *ip_address*

Example:

```
switch: set DEFAULT_ROUTER 192.168.2.1
```

d) Verify that you can ping the TFTP server **switch: ping** *ip_address_of_TFTP_server*

Example:

```
switch: ping 192.168.2.15
ping 192.168.2.1 with 32 bytes of data...
Host 192.168.2.1 is alive.
switch:
```

Step 7 From the bootloader prompt, initiate the boot command that assists you in recovering the software image on your switch.

WARNING: The emergency install command will erase your entire boot flash!

Alternatively, you can copy the image from TFTP to local flash through Telnet or Management port and then boot the device from local flash.

Preventing Autonegotiation Mismatches

The IEEE 802.3ab autonegotiation protocol manages the device settings for speed (10 Mb/s, 100 Mb/s, and 1000 Mb/s, excluding SFP module ports) and duplex (half or full). There are situations when this protocol can incorrectly align these settings, reducing performance. A mismatch occurs under these circumstances:

- A manually set speed or duplex parameter is different from the manually set speed or duplex parameter on the connected port.
- A port is set to autonegotiate, and the connected port is set to full duplex with no autonegotiation.

To maximize the device performance and ensure a link, follow one of these guidelines when changing the settings for duplex and speed:

- Let both ports autonegotiate both speed and duplex.
- Manually set the speed and duplex parameters for the ports on both ends of the connection.

Note If a remote device does not autonegotiate, configure the duplex settings on the two ports to match. The speed parameter can adjust itself even if the connected port does not autonegotiate.

Troubleshooting SFP Module Security and Identification

Cisco small form-factor pluggable (SFP) modules have a serial EEPROM that contains the module serial number, the vendor name and ID, a unique security code, and cyclic redundancy check (CRC). When an SFP module is inserted in the device, the device software reads the EEPROM to verify the serial number, vendor name and vendor ID, and recompute the security code and CRC. If the serial number, the vendor name or vendor ID, the security code, or CRC is invalid, the software generates a security error message and places the interface in an error-disabled state.

If you are using a non-Cisco SFP module, remove the SFP module from the device, and replace it with a Cisco module. After inserting a Cisco SFP module, use the **errdisable recovery cause gbic-invalid** global configuration command to verify the port status, and enter a time interval for recovering from the error-disabled state. After the elapsed interval, the device brings the interface out of the error-disabled state and retries the

operation. For more information about the **errdisable recovery** command, see the command reference for this release.

If the module is identified as a Cisco SFP module, but the system is unable to read vendor-data information to verify its accuracy, an SFP module error message is generated. In this case, you should remove and reinsert the SFP module. If it continues to fail, the SFP module might be defective.

Monitoring SFP Module Status

You can check the physical or operational status of an SFP module by using the **show interfaces transceiver** privileged EXEC command. This command shows the operational status, such as the temperature and the current for an SFP module on a specific interface and the alarm status. You can also use the command to check the speed and the duplex settings on an SFP module. For more information, see the **show interfaces transceiver** command in the command reference for this release.

Executing Ping

If you attempt to ping a host in a different IP subnetwork, you must define a static route to the network or have IP routing configured to route between those subnets.

IP routing is disabled by default on all devices.

Note Though other protocol keywords are available with the **ping** command, they are not supported in this release.

Use this command to ping another device on the network from the device:

Command	Purpose
ping ip *host* | *address* `Device# ping 192.168.52.3`	Pings a remote host through IP or by supplying the hostname or network address.

Monitoring Temperature

The Device monitors the temperature conditions and uses the temperature information to control the fans.

Monitoring the Physical Path

You can monitor the physical path that a packet takes from a source device to a destination device by using one of these privileged EXEC commands:

Table 16: Monitoring the Physical Path

Command	Purpose
tracetroute mac [**interface** *interface-id*] {*source-mac-address*} [**interface** *interface-id*] {*destination-mac-address*} [**vlan** *vlan-id*] [**detail**]	Displays the Layer 2 path taken by the packets from the specified source MAC address to the specified destination MAC address.

Command	Purpose
tracetroute mac ip {*source-ip-address* | *source-hostname*} {*destination-ip-address* | *destination-hostname*} [**detail**]	Displays the Layer 2 path taken by the packets from the specified source IP address or hostname to the specified destination IP address or hostname.

Executing IP Traceroute

Note Though other protocol keywords are available with the **traceroute** privileged EXEC command, they are not supported in this release.

Command	Purpose
traceroute ip *host* `Device# traceroute ip 192.51.100.1`	Traces the path that packets take through the network.

Running TDR and Displaying the Results

To run TDR, enter the **test cable-diagnostics tdr interface** *interface-id* privileged EXEC command.

To display the results, enter the **show cable-diagnostics tdr interface** *interface-id* privileged EXEC command.

Redirecting Debug and Error Message Output

By default, the network server sends the output from **debug** commands and system error messages to the console. If you use this default, you can use a virtual terminal connection to monitor debug output instead of connecting to the console port .

Possible destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server. The syslog format is compatible with 4.3 Berkeley Standard Distribution (BSD) UNIX and its derivatives.

Note Be aware that the debugging destination you use affects system overhead. When you log messages to the console, very high overhead occurs. When you log messages to a virtual terminal, less overhead occurs. Logging messages to a syslog server produces even less, and logging to an internal buffer produces the least overhead of any method.

For more information about system message logging, see *Configuring System Message Logging*.

Using the show platform Command

The output from the **show platform** privileged EXEC command provides some useful information about the forwarding results if a packet entering an interface is sent through the system. Depending upon the parameters entered about the packet, the output provides lookup table results and port maps used to calculate forwarding destinations, bitmaps, and egress information.

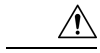
Most of the information in the output from the command is useful mainly for technical support personnel, who have access to detailed information about the device application-specific integrated circuits (ASICs). However, packet forwarding information can also be helpful in troubleshooting.

Using the show debug command

The **show debug** command is entered in privileged EXEC mode. This command displays all debug options available on the switch.

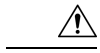
To view all conditional debug options run the command **show debug condition** The commands can be listed by selecting either a condition identifier *<1-1000>* or *all* conditions.

To disable debugging, use the **no debug all** command.



Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

Configuring OBFL



Caution We recommend that you do not disable OBFL and that you do not remove the data stored in the flash memory.

Troubleshooting Packet Loss

If your system exhibits partial or full loss of network connectivity or packet loss, perform basic troubleshooting procedures to eliminate the common causes. The common causes include:

- Bad cabling
- A bad port
- Speed and Duplex mismatch
- Network interface card (NIC) issues

1. If you troubleshoot these common reasons and you are not able to narrow down the problem, enter the **show platform hardware iomd 1/0 data-path** command to check the packet loss. If there are symptoms of packet loss, enter the **reload** command to soft reset the switch.
2. If the reload results in supervisor module diagnostic failure, power cycle the switch.
3. Enter the Generic On Line Diagnostics (GOLD) **show diagnostic bootup** command to determine if diagnostics fail.

   If diagnostics fail again, the problem is most likely the hardware.

   Contact Cisco Technical Support for further assistance.

4. If the supervisor module passes the diagnostic tests without any failure after the power cycle in Step 2, perform these steps:
   a. Collect the output from the **show tech-support** command.
   b. Remove all power supplies from the box, and collect the serial numbers, Cisco part number, and manufacturer of the power supplies.
   c. Contact Cisco Technical Support with the information that you collected.

Troubleshooting Interface Problems

If you see an error mentioned in the output of the command, **show interface** command, the reason could be:

- A physical layer problem, such as a faulty cable or NIC
- A configuration problem, such as a speed and duplex mismatch
- A performance problem, such as an oversubscription.

To understand and troubleshoot thses problems, refer the *Troubleshooting Switch Port and Interface Problems* at http://www.cisco.com/en/US/products/hw/switches/ps708/products_tech_note09186a008015bfd6.shtml

Troubleshooting when a Workstation Is Unable to Log In to the Network

If you observe that a workstation is unable to log into the network during startup or unable to obtain the DHCP address when you have powered up a client machine or rebooted, an initial connectivity delay that the switch introduced could be the problem. To verify this, check the following:

- Microsoft network client displays "No Domain Controllers Available".
- DHCP reports "No DHCP Servers Available".
- A Novell Internetwork Packet Exchange (IPX) network workstation does not have the Novell login screen upon bootup.
- An AppleTalk network client displays, "Access to your AppleTalk network has been interrupted. In order to reestablish your connection, open and close the AppleTalk control panel." The AppleTalk client chooser application can either fail to display a zone list or display an incomplete zone list.
- IBM Network stations can have one of these messages:
  - NSB83619—Address resolution failed
  - NSB83589—Failed to boot after 1 attempt
  - NSB70519—Failed to connect to a server

The reason for these symptoms can be an interface delay that either Spanning Tree Protocol (STP), EtherChannel, trunking, or an autonegotiation delay causes.

Verifying Troubleshooting of the Software Configuration

Displaying OBFL Information

Example: Verifying the Problem and Cause for High CPU Utilization

To determine if high CPU utilization is a problem, enter the **show processes cpu sorted** privileged EXEC command. Note the underlined information in the first line of the output example.

```
Device# show processes cpu sorted
CPU utilization for five seconds: 8%/0%; one minute: 7%; five minutes: 8%
PID Runtime(ms) Invoked uSecs 5Sec 1Min 5Min TTY Process
309 42289103 752750 56180 1.75% 1.20% 1.22% 0 RIP Timers
140 8820183 4942081 1784 0.63% 0.37% 0.30% 0 HRPC qos request
100 3427318 16150534 212 0.47% 0.14% 0.11% 0 HRPC pm-counters
192 3093252 14081112 219 0.31% 0.14% 0.11% 0 Spanning Tree
143 8 37 216 0.15% 0.01% 0.00% 0 Exec
...
<output truncated>
```

This example shows normal CPU utilization. The output shows that utilization for the last 5 seconds is 8%/0%, which has this meaning:

- The total CPU utilization is 8 percent, including both time running Cisco IOS processes and time spent handling interrupts.
- The time spent handling interrupts is zero percent.

Table 17: Troubleshooting CPU Utilization Problems

Type of Problem	Cause	Corrective Action
Interrupt percentage value is almost as high as total CPU utilization value.	The CPU is receiving too many packets from the network.	Determine the source of the network packet. Stop the flow, or change the switch configuration. See the section on “Analyzing Network Traffic.”
Total CPU utilization is greater than 50% with minimal time spent on interrupts.	One or more Cisco IOS process is consuming too much CPU time. This is usually triggered by an event that activated the process.	Identify the unusual event, and troubleshoot the root cause. See the section on “Debugging Active Processes.”

Configuration Examples for Troubleshooting Software

Example: Pinging an IP Host

This example shows how to ping an IP host:

```
Device# ping 192.168.52.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echoes to 192.168.52.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
Device#
```

Table 18: Ping Output Display Characters

Character	Description
!	Each exclamation point means receipt of a reply.
.	Each period means the network server timed out while waiting for a reply.
U	A destination unreachable error PDU was received.
C	A congestion experienced packet was received.
I	User interrupted test.
?	Unknown packet type.
&	Packet lifetime exceeded.

To end a ping session, enter the escape sequence (**Ctrl-^ X** by default). Simultaneously press and release the **Ctrl**, **Shift**, and **6** keys and then press the **X** key.

Example: Performing a Traceroute to an IP Host

This example shows how to perform a **traceroute** to an IP host:

```
Device# traceroute ip 192.168.2.10

Type escape sequence to abort.
Tracing the route to 192.168.2.10

  1 192.168.2.1 0 msec 0 msec 4 msec
  2 192.168.2.203 12 msec 8 msec 0 msec
  3 192.168.2.100 4 msec 0 msec 0 msec
  4 192.168.2.10 0 msec 4 msec 0 msec
```

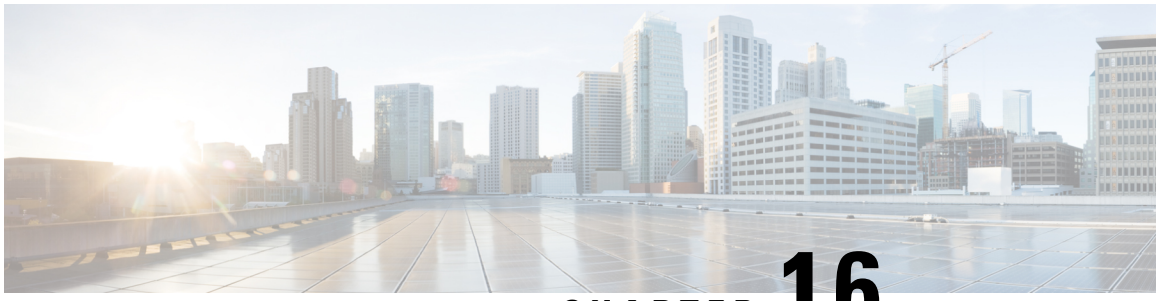
The display shows the hop count, the IP address of the router, and the round-trip time in milliseconds for each of the three probes that are sent.

Table 19: Traceroute Output Display Characters

Character	Description
*	The probe timed out.
?	Unknown packet type.

Character	Description
A	Administratively unreachable. Usually, this output means that an access list is blocking traffic.
H	Host unreachable.
N	Network unreachable.
P	Protocol unreachable.
Q	Source quench.
U	Port unreachable.

To end a trace in progress, enter the escape sequence (**Ctrl-^ X** by default). Simultaneously press and release the **Ctrl**, **Shift**, and **6** keys and then press the **X** key.



CHAPTER 16

Reset and Device Zeroization

This section contains the following:

Device Zeroization

Zeroization consists of erasing any and all potentially sensitive information in the device. This includes erasure of Main memory, cache memories, and other memories containing packet data, NVRAM, and Flash memory. The process of zeroization is launched upon the initiation of a user command and a subsequent trigger.

Important **service declassify erase-nvram** is NOT guaranteed to securely and completely erase the data from the underlying file system. The data may be recoverable by forensic analysis techniques. Consider using **service declassify erase-all** to securely delete all data on the device

Important IOS cannot securely erase an SD Card, so integrators that want secure erasure must not include the SD Card.

By default, the device will have the zeroization feature disabled. SPI: Flash, I2C, mSATA SSD and ACT2 are not impacted by this feature.

Note Ensure that you are familiar with the Emergency Recovery Installation procedure BEFORE attempting to test the Zeroize feature.

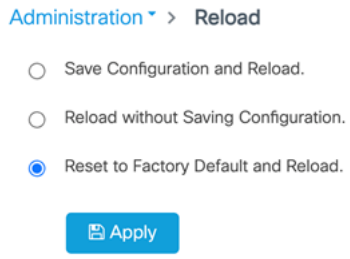
When zeroization is functionally active, the SYS LED indicates blinking yellow until the device reloads.

WARNING!

The CLI **service declassify erase-all** is literally a **software self-destruct mechanism** intended for defense and intelligence environments that attempts to wipe clean, all of the writable non-volatile storage on the device

to clear the device configuration, other stored configurations and all security credentials including any additional license keys.

Please do not use this feature in lieu of doing a **write erase** from the CLI or from the Administration page, Reload option of the WebUI. Invoke the reload with the **Reset to Factory Default and Reload** option and click **Apply**. See the following figure.



If **service declassify erase-all** is invoked, after restoring the IOS-XE image and device configuration, you must re-license the device using the standard Cisco Smart Licensing procedures which ultimately require a Cisco Smart Account and access to the internet or a satellite license server.

Push Button

The term Reset Button does not have the same meaning as with other devices. There is no actual button on the device, and the system integrator must configure their platform with a push button. Reset on a device does not cause the device to reboot, but initiates the configured level of Zeroization.

Zeroization can be triggered by the push button, or software-triggered by a privilege 15 user with console access. There is no remote access for security reasons. On triggering zeroization, the eMMC, NVRAM will be erased completely.

The zeroization process starts as soon as the push button is pressed down or the command is triggered. The CLI command, **service declassify**, is used to set the desired action in response to push button press. To prevent accidental erasure of the system configuration/image, the default setting is set to **no service declassify**.

Note While Cisco IOS and Cisco IOS-XE use the command line text of "declassify" in the command line interface (CLI) to enable the zeroize feature, in no way does this represent any specific endorsement or acknowledgment of a Government approved flash erasure methodology. Device Zeroization Declassification procedures are unique to each Government organization. Cisco solely provides the technical detail of the erasure operation here, not the policy distinction or any specific recommendation per classification. Please refer to your respective Government Agency policies, procedures, and recommendations for the handling of sensitive data to see if this procedure meets with those requirements.

There is a zeroization function available on the device when the system integrator has configured their platform with a push button.

- When the system is running in IOS mode, pressing this push button for 4+ seconds will cause files erase in flash, and will reset to factory-default mode on boot up.
- The button must be pressed while the system is turned on at the same time.
- The push button must continue to be held for more than 4 seconds after the power is turned on.

- Config-reg setting is in NVRAM, not changed by the push button.
- Pressing the push button when in rommon mode has no effect.
- Pressing the push button when in IOS mode causes a syslog message to appear and triggers a reload.
- Pressing the push button for more than 4+ seconds after power up displays the following message when reset has been triggered:

```
System Bootstrap, Version 1.4(DEV) [vandvisw-vandvisw 113], DEVELOPMENT SOFTWARE
Copyright (c) 1994-2019 by cisco Systems, Inc.
Compiled at Mon Jun 3 10:56:19 2019 by vandvisw
ESS-9300-CON-K9 platform with 4194304 Kbytes of main memory
MCU Version - Bootloader: 8, App: 10
MCU is in application mode.
Reset button push detected
```

Microcontroller Unit (MCU)

The MCU is part of the device hardware. It performs the following functions:

- Monitors the Push button status at power up
- Monitors the system hardware watchdog output
- Maintains Reset Reason register
- Controls the SYS LED

The MCU versions are displayed using show version. Details on MCU version and upgrade status are also stored in Flash: as boothelper.log. The MCU is automatically upgraded by the software.

```
Router#show ver | i MCU
MCU bootloader version: 8
MCU application version: 10
Router#cat flash:boothelper.log
Logging at Fri Nov 15 05:00:54 Universal 2019
boot loader upgrade enabled
Bootloader is up-to-date
Current MCU App version is 10
MCU firmware is up-to-date
```

In the event the MCU Application is corrupt, or does not match the Release Notes version, this has to be repaired. Steps to recover from this state: Reload router, hit Ctrl+C to break into rommon mode.

```
Rommon>set MCU_UPGRADE=IGNORE
 –
Ignore MCU firmware upgrade errors
.
Rommon>sync
Rommon>reset
Rommon>boot bootflash:<image>
```

Once the MCU successfully upgrades, you can disable/unset this IGNORE option in rommon. Details on other MCU setting rommon options follow: (there are no available IOS configuration options or linux shell mode troubleshooting measures)

```
set MCU_UPGRADE=SKIP
 –
Prevents MCU firmware upgrade from taking place
.
```

```
set MCU_UPGRADE=FORCE
 –
Forces MCU firmware upgrade to take place
.
unset MCU_UPGRADE
 –
Normal operation. Allows automatic upgrade
.
```

Zeroization Trigger

Zeroization can be triggered by either software or by the push button. In either case, there are a series of commands that need to be entered.

```
Router#config terminal
Router(config)#service declassify {erase-nvram | erase-all}
```

To confirm if service declassify is enabled:

```
Router#show declassify

Declassify facility: Enabled=Yes  In Progress=No
                     Erase flash=Yes  Erase nvram=Yes
  Declassify Console and Aux Ports
  Shutdown Interfaces
  Reload system
```

To remove declassification, use the following command:

```
Router(config)#no service declassify
```

To Trigger Zeroization

To trigger the zeroization from the command line:

```
Router#declassify trigger
```

To trigger the zeroization from the push button, press and hold the button for 4+ seconds. When the system auto reloads, it will come up in ROMMON mode: "$$" with bootflash: wiped clean.

Command Line Interface

There are two levels of zeroization actions, erase-nvram and erase-all. The following CLI shows the options:

```
router(config)#service declassify ?
erase-nvram    Enable erasure of router configuration as declassification action. Default
is no erasure.
erase-all      Enable erasure of both flash and nvram file systems as part of
declassification. Default is no erasure
```

The “erase-nvram” level of declassification process searches for the following files, and erases the ones found.

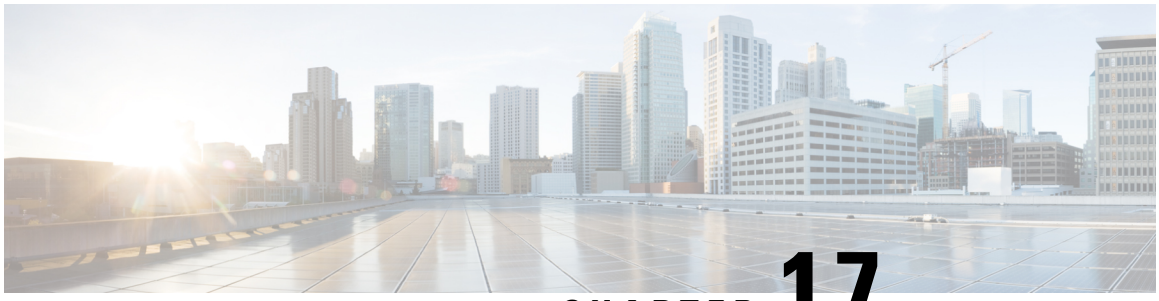
- flash:/nvram_config
- flash:/vlan.dat

This also erases the complete NVRAM filesystem, therefore, all configurations, including startup and running configurations will get deleted.

The “erase-all” level of zeroization process erases the entire flash file system. This also wipes out all files and perma-locked bootable image(s). All interfaces are shut down before this process. Here, erasure of individual files in the flash file system is not possible and the only option is to erase the entire flash file system. This also erases packet data, ASIC data and processors related caches along with scrubbing Main memory.

With any level of zeroization, the router always fall back to the ROMMON prompt on the console after the erasure of configuration files or flash file system.

Caution The device does not support USB hot plug when it is in ROMMON mode.



CHAPTER 17

Additional Information and Configuration Guides

This chapter contains the following sections:

Additional Information

The ESS9300 offers a rich IOS-XE feature set. This marketing data sheet provides a complete list of all of the features.

Previous chapters in this guide provided an introduction the the ESS9300, as well as some of the basic configuration and feature differences for this product. The IOS-XE Operating System runs on numerous switching devices, and as such, has a wealth of additional configuration information.

The following is a sample of additional resources to use:

- Cisco Catalyst Rugged Series Industrial Ethernet Switches
- Cisco IOS XE

Additional Configuration Guides

The following links provide access to other Cisco IOS-XE switching configuration guides. Be sure to read the previous content in this guide as well as the marketing data sheet to understand what features are available on the ESS9300.

- CIP and MODBUS Configuration Guide
- IP Routing Configuration Guide
- IP Multicast Routing Configuration Guide
- Layer 2 Configuration Guide
- Network Management Configuration Guide
- PROFINET Configuration Guide

- QoS Configuration Guide
- Redundancy Protocol Configuration Guide
- Security Configuration Guide
- System Management Configuration Guide

Communications, Services, and Additional Information

- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business impact you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class apps, products, solutions, and services, visit Cisco DevNet.
- To obtain general networking, training, and certification titles, visit Cisco Press.
- To find warranty information for a specific product or product family, access Cisco Warranty Finder.

Cisco Bug Search Tool

Cisco Bug Search Tool (BST) is a gateway to the Cisco bug-tracking system, which maintains a comprehensive list of defects and vulnerabilities in Cisco products and software. The BST provides you with detailed defect information about your products and software.

Documentation Feedback

To provide feedback about Cisco technical documentation, use the feedback form available in the right pane of every online document.