Performing Device Setup Configuration

- Restrictions for Performing Device Setup Configuration, on page 1
- Information About Performing Device Setup Configuration, on page 1
- How to Perform Device Setup Configuration, on page 11
- Configuration Examples for Device Setup Configuration, on page 20
- Additional References For Performing Device Setup, on page 37
- Feature History for Performing Device Setup Configuration, on page 37

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

To start your device, you need to follow the procedures described in the *Cisco Catalyst 9500 Series Switches Hardware Installation Guide* for installing and powering on the device and setting up the initial device configuration.

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 also performed:

  Cisco Catalyst 9500 Series Switches:

- Power over Ethernet (PoE) controller functionality test to check the chip accessibility, firmware download, and health status of the power-sourcing equipment.
- Thermal test to verify the temperature reading from the device sensor.
- Fan modules test to verify if all the inserted fan modules are working properly on the board.
- Federal Information Processing Standard (FIPS) MACsec test.

Cisco Catalyst 9500 Series High Performance Switches:

- MAC loopback test to verify the data path between the CPU and network ports connected to each module. If this test fails for any of the ports, the ports are forced into error-disabled state, and the module is marked as *post-fail* in the **show module** command output.

For information about the complete list of supported online diagnostics, see the Configuring Online Diagnostics chapter.

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.

If the switch experiences 5 consecutive unexpected reloads within 15 minutes of startup, autoboot will be disabled, and the switch will enter ROMMON mode. To recover, issue the **boot** command manually. This prevents continuous boot loops and ensures system stability.

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, Software Maintenance Upgrade (SMU), In-Service Software Upgrade (ISSU)and In-Service Model Update (data model package).

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 on a standalone switch.
- Software rollback to a previously installed package set.

Software Boot Modes

Your supports two modes to boot the software packages:

Installed Boot Mode

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

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

Note We recommend that you use the install mode for Cisco Catalyst 9200 Series Switches.

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.

Bundle Boot Mode

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

```
switch:  boot flash:cat9k_iosxe.16.05.01a.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:

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:cat9k_2.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**, **install activate**, and **install commit** 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.

Note The install operation through SCP is not supported.

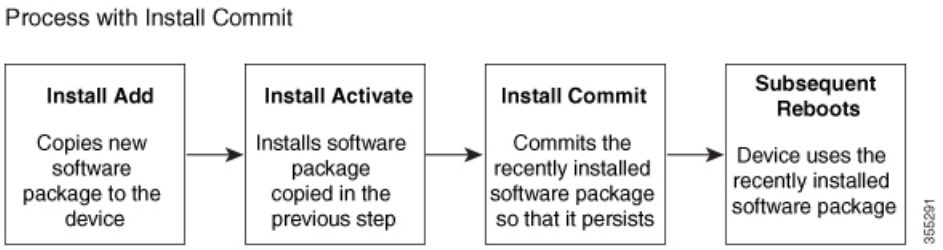
For the **install activate** command to work, the package must be available in the device bootflash. When this command is configured, previously added packages from the .bin file get activated, and the system reloads.

Enable the **install commit** command to make updates persistent over reloads.

Installing an update replaces any previously installed software image. At any time, only one image is installed on the device.

The following flow chart explains how the software install works:

Figure 1: Committing a Software Package



Note The **install activate** command reloads the device with the new image.

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. For a new switch, enter a new password for 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 1.

Default Switch Information

Table 1: 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.

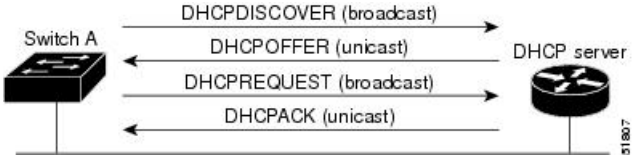
DHCP-based autoconfiguration replaces the BOOTP client functionality on your device.

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 2: 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.

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. Unplug the device power cord, and press the **Mode** button while reconnecting the power cord. You can release the **Mode** button after the system LED begins flashing green and remains solid. The boot loader device prompt then appears.

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.

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 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:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
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:** `Device(dhcp-config)# exit`	Returns to global configuration mode.
Step 8	**tftp-server flash**:*filename.text* **Example:** `Device(config)# tftp-server`	Specifies the configuration file on the TFTP server.

	Command or Action	Purpose
	`flash:config-boot.text`	
Step 9	**interface** *interface-id* **Example:** `Device(config)# interface fortygigabitethernet1/0/4`	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.

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.

	Command or Action	Purpose
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 10.10.10.2 255.255.255.0`	Enters the IP address and subnet mask.
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:**	Displays the interfaces status for the specified VLAN.

	Command or Action	Purpose
	`Device# show interfaces vlan 99`	
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.

	Command or Action	Purpose
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.

Booting the Device in Installed Mode

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.
Step 2	**install add file tftp:** *filename* [**activate commit**] **Example:** `Device# install add file tftp://172.16.0.1//tftpboot/folder1/cat9k_iosxe.16.06.01.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.

	Command or Action	Purpose
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/cat9k_iosxe.16.06.01.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.
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.

	Command or Action	Purpose
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.

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:`
	`switch:BOOT=tftp://10.0.0.2/cat9k_iosxe.16.05.01a.SPA.bin`
	`switch:`
	`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.

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.16.05.01a.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 [Everest], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 16.5.1a,
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 C9300-48P (X86) processor with 818597K/6147K bytes of memory.
Processor board ID FCW2049G03S
2048K bytes of non-volatile configuration memory.
8388608K bytes of physical memory.
1638400K bytes of Crash Files at crashinfo:.
11264000K bytes of Flash at flash:.
0K bytes of WebUI ODM Files at webui:.

Base Ethernet MAC Address          : 04:6c:9d:01:3b:80
Motherboard Assembly Number        : 73-17956-04
Motherboard Serial Number          : FOC20465ABU
Model Revision Number              : P4B
Motherboard Revision Number        : 04
Model Number                       : C9300-48P
System Serial Number               : FCW2049G03S


%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_iosxe.16.05.01a.SPA.bin

Attempting to boot from [flash:cat9k_iosxe.16.05.01a.SPA.bin]
Located cat9k_iosxe.16.05.01a.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 [Everest], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 16.5.1a,
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 C9300-24U (X86) processor with 818597K/6147K bytes of memory.
Processor board ID FCW2111G00X
2048K bytes of non-volatile configuration memory.
8388608K bytes of physical memory.
1638400K bytes of Crash Files at crashinfo:.
11264000K bytes of Flash at flash:.
15633392K bytes of USB Flash at usbflash0:.
0K bytes of WebUI ODM Files at webui:.
```

```
Base Ethernet MAC Address          : 04:6c:9d:1e:2a:80
Motherboard Assembly Number        : 73-17954-05
Motherboard Serial Number          : FOC21094MWL
Model Revision Number              : PP
Motherboard Revision Number        : 05
Model Number                       : C9300-24U
System Serial Number               : FCW2111G00X


%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: Managing an Update Package

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

```
Device# install add file flash:cat9k_iosxe.16.06.02.SPA.bin activate commit

install_add_activate_commit: START Mon Oct 30 19:54:51 UTC 2017

System configuration has been modified.
Press Yes(y) to save the configuration and proceed.
Press No(n) for proceeding without saving the configuration.
Press Quit(q) to exit, you may save configuration and re-enter the command. [y/n/q]y
Building configuration...

[OK]Modified configuration has been saved

*Oct 30 19:54:55.633: %IOSXE-5-PLATFORM: Switch 1 R0/0: Oct 30 19:54:55 install_engine.sh:

%INSTALL-5-INSTALL_START_INFO: Started install one-shot
flash:cat9k_iosxe.16.06.02.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 ---
Info: Finished copying flash:cat9k_iosxe.16.06.02.SPA.bin to the selected switch(es)
Finished initial file syncing

--- Starting Add ---
Performing Add on all members
  [1] Add package(s) on switch 1
  [1] Finished Add on switch 1
Checking status of Add on [1]
Add: Passed on [1]
Finished Add

install_add_activate_commit: Activating PACKAGE
Following packages shall be activated:
/flash/cat9k-wlc.16.06.02.SPA.pkg
/flash/cat9k-webui.16.06.02.SPA.pkg
/flash/cat9k-srdriver.16.06.02.SPA.pkg
/flash/cat9k-sipspa.16.06.02.SPA.pkg
/flash/cat9k-sipbase.16.06.02.SPA.pkg
```

```
/flash/cat9k-rpboot.16.06.02.SPA.pkg
/flash/cat9k-rpbase.16.06.02.SPA.pkg
/flash/cat9k-guestshell.16.06.02.SPA.pkg
/flash/cat9k-espbase.16.06.02.SPA.pkg
/flash/cat9k-cc_srdriver.16.06.02.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
  [1] Activate package(s) on switch 1
  [1] Finished Activate on switch 1
Checking status of Activate on [1]
Activate: Passed on [1]
Finished Activate

--- Starting Commit ---
Performing Commit on all members

*Oct 30 19:57:41.145: %IOSXE-5-PLATFORM: Switch 1 R0/0: Oct 30 19:57:41 rollback_timer.sh:

%INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Install auto abort timer will expire in 7200
 seconds  [1]
Commit package(s) on switch 1
  [1] Finished Commit on switch 1
Checking status of Commit on [1]
Commit: Passed on [1]
Finished Commit

Install will reload the system now!
SUCCESS: install_add_activate_commit  Mon Oct 30 19:57:48 UTC 2017

Device#
*Oct 30 19:57:48.384: %IOSXE-5-PLATFORM: Switch 1 R0/0: Oct 30 19:57:48 install_engine.sh:

%INSTALL-5-INSTALL_COMPLETED_INFO: Completed install one-shot PACKAGE
flash:cat9k_iosxe.16.06.02.SPA.bin

 Chassis 1 reloading, reason - Reload command
```

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

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

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:

```
Device# install commit
install_commit: START Fri Jun 23 21:24:45 IST 2017
install_commit: Committing PACKAGE

--- Starting Commit ---
Performing Commit on Active/Standby
  [R0] Commit package(s) on R0
  [R0] Finished Commit on R0
Checking status of Commit on [R0]
Commit: Passed on [R0]
Finished Commit

SUCCESS: install_commit  Fri Jun 23 21:24:48 IST 2017

Device#
```

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

```
Device# install rollback to committed

install_rollback: START Mon Oct 30 20:53:33 UTC 2017

This operation requires a reload of the system. Do you want to proceed? [y/n]

*Oct 30 20:53:34.713: %IOSXE-5-PLATFORM: Switch 1 R0/0: Oct 30 20:53:34
install_engine.sh: %INSTALL-5-INSTALL_START_INFO: Started install rollback

--- Starting Rollback ---
Performing Rollback on all members
  [1] Rollback package(s) on switch 1
    --- Starting rollback impact ---
    Changes that are part of this rollback
    Current    :  rp 0 0   rp_boot         cat9k-rpboot.16.06.02.prd9.SPA.pkg
    Current    :  rp 1 0   rp_boot         cat9k-rpboot.16.06.02.prd9.SPA.pkg
    Replacement:  rp 0 0   rp_boot         cat9k-rpboot.16.06.02.SPA.pkg
    Replacement:  rp 1 0   rp_boot         cat9k-rpboot.16.06.02.SPA.pkg
    Current    :   cc 0  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.prd9.SPA.pkg
    Current    :   cc 0 0   cc              cat9k-sipbase.16.06.02.prd9.SPA.pkg
    Current    :   cc 0 0   cc_spa          cat9k-sipspa.16.06.02.prd9.SPA.pkg
    Current    :   cc 1  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.prd9.SPA.pkg
    Current    :   cc 1 0   cc              cat9k-sipbase.16.06.02.prd9.SPA.pkg
    Current    :   cc 1 0   cc_spa          cat9k-sipspa.16.06.02.prd9.SPA.pkg
    Current    :   cc 10 0  cc              cat9k-sipbase.16.06.02.prd9.SPA.pkg
    Current    :   cc 10 0  cc_spa          cat9k-sipspa.16.06.02.prd9.SPA.pkg
    Current    :   cc 10 0  cc_srdriver     cat9k-cc_srdriver.16.06.02.prd9.SPA.pkg
    Current    :   cc 2  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.prd9.SPA.pkg
    Current    :   cc 2 0   cc              cat9k-sipbase.16.06.02.prd9.SPA.pkg
    Current    :   cc 2 0   cc_spa          cat9k-sipspa.16.06.02.prd9.SPA.pkg
    Current    :   cc 3  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.prd9.SPA.pkg
    Current    :   cc 3 0   cc              cat9k-sipbase.16.06.02.prd9.SPA.pkg
    Current    :   cc 3 0   cc_spa          cat9k-sipspa.16.06.02.prd9.SPA.pkg
    Current    :   cc 4  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.prd9.SPA.pkg
```

```
Current    :   cc 4 0   cc              cat9k-sipbase.16.06.02.prd9.SPA.pkg
Current    :   cc 4 0   cc_spa          cat9k-sipspa.16.06.02.prd9.SPA.pkg
Current    :   cc 5  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.prd9.SPA.pkg
Current    :   cc 5 0   cc              cat9k-sipbase.16.06.02.prd9.SPA.pkg
Current    :   cc 5 0   cc_spa          cat9k-sipspa.16.06.02.prd9.SPA.pkg
Current    :   cc 6  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.prd9.SPA.pkg
Current    :   cc 6 0   cc              cat9k-sipbase.16.06.02.prd9.SPA.pkg
Current    :   cc 6 0   cc_spa          cat9k-sipspa.16.06.02.prd9.SPA.pkg
Current    :   cc 7  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.prd9.SPA.pkg
Current    :   cc 7 0   cc              cat9k-sipbase.16.06.02.prd9.SPA.pkg
Current    :   cc 7 0   cc_spa          cat9k-sipspa.16.06.02.prd9.SPA.pkg
Current    :   cc 8  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.prd9.SPA.pkg
Current    :   cc 8 0   cc              cat9k-sipbase.16.06.02.prd9.SPA.pkg
Current    :   cc 8 0   cc_spa          cat9k-sipspa.16.06.02.prd9.SPA.pkg
Current    :   cc 9  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.prd9.SPA.pkg
Current    :   cc 9 0   cc              cat9k-sipbase.16.06.02.prd9.SPA.pkg
Current    :   cc 9 0   cc_spa          cat9k-sipspa.16.06.02.prd9.SPA.pkg
Current    :   fp 0 0   fp              cat9k-espbase.16.06.02.prd9.SPA.pkg
Current    :   fp 1 0   fp              cat9k-espbase.16.06.02.prd9.SPA.pkg
Current    :   rp 0 0   guestshell      cat9k-guestshell.16.06.02.prd9.SPA.pkg
Current    :   rp 0 0   rp_base         cat9k-rpbase.16.06.02.prd9.SPA.pkg
Current    :   rp 0 0   rp_daemons      cat9k-rpbase.16.06.02.prd9.SPA.pkg
Current    :   rp 0 0   rp_iosd         cat9k-rpbase.16.06.02.prd9.SPA.pkg
Current    :   rp 0 0   rp_security     cat9k-rpbase.16.06.02.prd9.SPA.pkg
Current    :   rp 0 0   rp_webui        cat9k-webui.16.06.02.prd9.SPA.pkg
Current    :   rp 0 0   rp_wlc          cat9k-wlc.16.06.02.prd9.SPA.pkg
Current    :   rp 0 0   srdriver        cat9k-srdriver.16.06.02.prd9.SPA.pkg
Current    :   rp 1 0   guestshell      cat9k-guestshell.16.06.02.prd9.SPA.pkg
Current    :   rp 1 0   rp_base         cat9k-rpbase.16.06.02.prd9.SPA.pkg
Current    :   rp 1 0   rp_daemons      cat9k-rpbase.16.06.02.prd9.SPA.pkg
Current    :   rp 1 0   rp_iosd         cat9k-rpbase.16.06.02.prd9.SPA.pkg
Current    :   rp 1 0   rp_security     cat9k-rpbase.16.06.02.prd9.SPA.pkg
Current    :   rp 1 0   rp_webui        cat9k-webui.16.06.02.prd9.SPA.pkg
Current    :   rp 1 0   rp_wlc          cat9k-wlc.16.06.02.prd9.SPA.pkg
Current    :   rp 1 0   srdriver        cat9k-srdriver.16.06.02.prd9.SPA.pkg
Replacement:   cc 0  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
Replacement:   cc 0 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
Replacement:   cc 0 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
Replacement:   cc 1  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
Replacement:   cc 1 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
Replacement:   cc 1 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
Replacement:   cc 10 0  cc              cat9k-sipbase.16.06.02.SPA.pkg
Replacement:   cc 10 0  cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
Replacement:   cc 10 0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
Replacement:   cc 2  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
Replacement:   cc 2 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
Replacement:   cc 2 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
Replacement:   cc 3  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
Replacement:   cc 3 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
Replacement:   cc 3 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
Replacement:   cc 4  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
Replacement:   cc 4 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
Replacement:   cc 4 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
Replacement:   cc 5  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
Replacement:   cc 5 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
Replacement:   cc 5 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
Replacement:   cc 6  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
Replacement:   cc 6 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
Replacement:   cc 6 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
Replacement:   cc 7  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
Replacement:   cc 7 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
Replacement:   cc 7 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
Replacement:   cc 8  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
Replacement:   cc 8 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
```

```
    Replacement:   cc 8 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   cc 9  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
    Replacement:   cc 9 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
    Replacement:   cc 9 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   fp 0 0   fp              cat9k-espbase.16.06.02.SPA.pkg
    Replacement:   fp 1 0   fp              cat9k-espbase.16.06.02.SPA.pkg
    Replacement:   rp 0 0   guestshell      cat9k-guestshell.16.06.02.SPA.pkg
    Replacement:   rp 0 0   rp_base         cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 0 0   rp_daemons      cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 0 0   rp_iosd         cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 0 0   rp_security     cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 0 0   rp_webui        cat9k-webui.16.06.02.SPA.pkg
    Replacement:   rp 0 0   rp_wlc          cat9k-wlc.16.06.02.SPA.pkg
    Replacement:   rp 0 0   srdriver        cat9k-srdriver.16.06.02.SPA.pkg
    Replacement:   rp 1 0   guestshell      cat9k-guestshell.16.06.02.SPA.pkg
    Replacement:   rp 1 0   rp_base         cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 1 0   rp_daemons      cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 1 0   rp_iosd         cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 1 0   rp_security     cat9k-rpbase.16.06.02.SPA.pkg

    Chassis 1 reloading, reason - Reload command

    Replacement:   rp 1 0   rp_webui        cat9k-webui.16.06.02.SPA.pkg
    Replacement:   rp 1 0   rp_wlc          cat9k-wlc.16.06.02.SPA.pkg
    Replacement:   rp 1 0   srdriver        cat9k-srdriver.16.06.02.SPA.pkg
    Finished rollback impact
  [1] Finished Rollback on switch 1
Checking status of Rollback on [1]
Rollback: Passed on [1]
Finished Rollback

Install will reload the system now!
SUCCESS: install_rollback  Mon Oct 30 20:54:23 UTC 2017

Device#
*Oct 30 20:54:23.576: %IOSXE-5-PLATFORM: Switch 1 R0/0: Oct 30 20:54:23
 install_engine.sh: %INSTALL-5-INSTALL_COMPLETED_INFO: Completed install rollback PACKAGE
*Oct 30 20:54:25.416: %STACKMGR-1-RELOAD: Switch 1 R0/0: stack_mgr:
 Reloading due to reason Reload command Oct 30 20:54:31.615 FP0/0: %PMAN-5-EXITACTION:
 Process manager is exiting: reload fp action requested
Oct 30 20:54
```

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

```
Device# install remove inactive

install_remove: START Mon Oct 30 19:51:48 UTC 2017
Cleaning up unnecessary package files
  Scanning boot directory for packages ... done.
  Preparing packages list to delete ...
  done.

The following files will be deleted:
[switch 1]:
/flash/cat9k-cc_srdriver.16.06.02.SPA.pkg
/flash/cat9k-espbase.16.06.02.SPA.pkg
/flash/cat9k-guestshell.16.06.02.SPA.pkg
/flash/cat9k-rpbase.16.06.02.SPA.pkg
/flash/cat9k-rpboot.16.06.02.SPA.pkg
/flash/cat9k-sipbase.16.06.02.SPA.pkg
/flash/cat9k-sipspa.16.06.02.SPA.pkg
/flash/cat9k-srdriver.16.06.02.SPA.pkg
/flash/cat9k-webui.16.06.02.SPA.pkg
/flash/cat9k-wlc.16.06.02.SPA.pkg
```

```
/flash/packages.conf

Do you want to remove the above files? [y/n]y
[switch 1]:
Deleting file flash:cat9k-cc_srdriver.16.06.02.SPA.pkg ... done.
Deleting file flash:cat9k-espbase.16.06.02.SPA.pkg ... done.
Deleting file flash:cat9k-guestshell.16.06.02.SPA.pkg ... done.
Deleting file flash:cat9k-rpbase.16.06.02.SPA.pkg ... done.
Deleting file flash:cat9k-rpboot.16.06.02.SPA.pkg ... done.
Deleting file flash:cat9k-sipbase.16.06.02.SPA.pkg ... done.
Deleting file flash:cat9k-sipspa.16.06.02.SPA.pkg ... done.
Deleting file flash:cat9k-srdriver.16.06.02.SPA.pkg ... done.
Deleting file flash:cat9k-webui.16.06.02.SPA.pkg ... done.
Deleting file flash:cat9k-wlc.16.06.02.SPA.pkg ... done.
Deleting file flash:packages.conf ... done.
SUCCESS: Files deleted.
--- Starting Post_Remove_Cleanup ---
Performing Post_Remove_Cleanup on all members
  [1] Post_Remove_Cleanup package(s) on switch 1
  [1] Finished Post_Remove_Cleanup on switch 1
Checking status of Post_Remove_Cleanup on [1]
Post_Remove_Cleanup: Passed on [1]
Finished Post_Remove_Cleanup

SUCCESS: install_remove  Mon Oct 30 19:52:25 UTC 2017
Device#
```

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

```
Device# install abort

/usr/binos/conf/chasutils.sh: line 428: chasfs_is_dominica: readonly function
/usr/binos/conf/chasutils.sh: line 428: chasfs_is_dominica: readonly function
/usr/binos/conf/chasutils.sh: line 428: chasfs_is_dominica: readonly function
/usr/binos/conf/chasutils.sh: line 428: chasfs_is_dominica: readonly function
install_abort: START Mon Oct 30 20:27:32 UTC 2017
install_abort: Abort type PACKAGE subtype NONE smutype NONE

This install abort would require a reload. Do you want to proceed? [y/n]

*Oct 30 20:27:33.189: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine: Started
 install abort
--- Starting Abort ---
Performing Abort on all members
/usr/binos/conf/chasutils.sh: line 428: chasfs_is_dominica: readonly function
/usr/binos/conf/chasutils.sh: line 428: chasfs_is_dominica: readonly function
/usr/binos/conf/chasutils.sh: line 428: chasfs_is_dominica: readonly function
/usr/binos/conf/chasutils.sh: line 428: chasfs_is_dominica: readonly function
/usr/binos/conf/chasutils.sh: line 428: chasfs_is_dominica: readonly function
/usr/binos/conf/chasutils.sh: line 428: chasfs_is_dominica: readonly function
  [1] Abort package(s) on switch 1
    --- Starting rollback impact ---
    Changes that are part of this rollback
    Current    :  rp 0 0   rp_boot
cat9k-rpboot.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :  rp 1 0   rp_boot
cat9k-rpboot.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Replacement:  rp 0 0   rp_boot         cat9k-rpboot.16.06.02.SPA.pkg
    Replacement:  rp 1 0   rp_boot         cat9k-rpboot.16.06.02.SPA.pkg
    Current    :   cc 0  0  cc_srdriver
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 0 0   cc
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 0 0   cc_spa
```

```
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 1  0  cc_srdriver
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 1 0   cc
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 1 0   cc_spa
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 10 0  cc
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 10 0  cc_spa
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 10 0  cc_srdriver
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 2  0  cc_srdriver
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 2 0   cc
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 2 0   cc_spa
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 3  0  cc_srdriver
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 3 0   cc
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 3 0   cc_spa
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 4  0  cc_srdriver
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 4 0   cc
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 4 0   cc_spa
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 5  0  cc_srdriver
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 5 0   cc
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 5 0   cc_spa
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 6  0  cc_srdriver
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 6 0   cc
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 6 0   cc_spa
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 7  0  cc_srdriver
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 7 0   cc
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 7 0   cc_spa
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 8  0  cc_srdriver
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 8 0   cc
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 8 0   cc_spa
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 9  0  cc_srdriver
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 9 0   cc
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   cc 9 0   cc_spa
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   fp 0 0   fp
cat9k-espbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   fp 1 0   fp
```

```
cat9k-espbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 0 0   guestshell
cat9k-guestshell.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 0 0   rp_base
cat9k-rpbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 0 0   rp_daemons
cat9k-rpbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 0 0   rp_iosd
cat9k-rpbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 0 0   rp_security
cat9k-rpbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 0 0   rp_webui
cat9k-webui.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 0 0   rp_wlc
cat9k-wlc.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 0 0   srdriver
cat9k-srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 1 0   guestshell
cat9k-guestshell.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 1 0   rp_base
cat9k-rpbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 1 0   rp_daemons
cat9k-rpbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 1 0   rp_iosd
cat9k-rpbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 1 0   rp_security
cat9k-rpbase.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 1 0   rp_webui
cat9k-webui.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 1 0   rp_wlc
cat9k-wlc.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Current    :   rp 1 0   srdriver
cat9k-srdriver.BLD_POLARIS_DEV_LATEST_20171029_082249.SSA.pkg
    Replacement:   cc 0  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
    Replacement:   cc 0 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
    Replacement:   cc 0 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   cc 1  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
    Replacement:   cc 1 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
    Replacement:   cc 1 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   cc 10 0  cc              cat9k-sipbase.16.06.02.SPA.pkg
    Replacement:   cc 10 0  cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   cc 10 0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
    Replacement:   cc 2  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
    Replacement:   cc 2 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
    Replacement:   cc 2 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   cc 3  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
    Replacement:   cc 3 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
    Replacement:   cc 3 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   cc 4  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
    Replacement:   cc 4 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
    Replacement:   cc 4 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   cc 5  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
    Replacement:   cc 5 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
    Replacement:   cc 5 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   cc 6  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
    Replacement:   cc 6 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
    Replacement:   cc 6 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   cc 7  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
    Replacement:   cc 7 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
    Replacement:   cc 7 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   cc 8  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
    Replacement:   cc 8 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
    Replacement:   cc 8 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   cc 9  0  cc_srdriver     cat9k-cc_srdriver.16.06.02.SPA.pkg
```

```
    Replacement:   cc 9 0   cc              cat9k-sipbase.16.06.02.SPA.pkg
    Replacement:   cc 9 0   cc_spa          cat9k-sipspa.16.06.02.SPA.pkg
    Replacement:   fp 0 0   fp              cat9k-espbase.16.06.02.SPA.pkg
    Replacement:   fp 1 0   fp              cat9k-espbase.16.06.02.SPA.pkg
    Replacement:   rp 0 0   guestshell      cat9k-guestshell.16.06.02.SPA.pkg
    Replacement:   rp 0 0   rp_base         cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 0 0   rp_daemons      cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 0 0   rp_iosd         cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 0 0   rp_security     cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 0 0   rp_webui        cat9k-webui.16.06.02.SPA.pkg
    Replacement:   rp 0 0   rp_wlc          cat9k-wlc.16.06.02.SPA.pkg
    Replacement:   rp 0 0   srdriver        cat9k-srdriver.16.06.02.SPA.pkg
    Replacement:   rp 1 0   guestshell      cat9k-guestshell.16.06.02.SPA.pkg
    Replacement:   rp 1 0   rp_base         cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 1 0   rp_daemons      cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 1 0   rp_iosd         cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 1 0   rp_security     cat9k-rpbase.16.06.02.SPA.pkg
    Replacement:   rp 1 0   rp_webui        cat9k-webui.16.06.02.SPA.pkg
    Replacement:   rp 1 0   rp_wlc          cat9k-wlc.16.06.02.SPA.pkg
    Replacement:   rp 1 0   srdriver        cat9k-srdriver.16.06.02.SPA.pkg
    Finished rollback impact
  [1] Finished Abort on switch 1
Checking status of Abort on [1]
Abort: Passed on [1]
Finished Abort

/usr/binos/conf/chasutils.sh: line 428: chasfs_is_dominica: readonly function
[1]: Performing MCU Upgrade Service
/usr/binos/conf/provfunc.sh: line 8792: $l_log_file: ambiguous redirect
  SUCCESS: MCU Upgrade Service finished
Install will reload the system now!
SUCCESS: install_abort  Mon Oct 30 20:28:21 UTC 2017
/usr/binos/conf/chasutils.sh: line 428: chasfs_is_dominica: readonly function
```

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

```
Device# install activate auto-abort-timer 30

install_activate: START Mon Oct 30 20:42:28 UTC 2017
install_activate: Activating PACKAGE

*Oct 30 20:42:29.149: %IOSXE-5-PLATFORM: Switch 1 R0/0: Oct 30 20:42:29 install_engine.sh:

%INSTALL-5-INSTALL_START_INFO: Started install activateFollowing packages shall be activated:
/flash/cat9k-wlc.16.06.02.prd9.SPA.pkg
/flash/cat9k-webui.16.06.02.prd9.SPA.pkg
/flash/cat9k-srdriver.16.06.02.prd9.SPA.pkg
/flash/cat9k-sipspa.16.06.02.prd9.SPA.pkg
/flash/cat9k-sipbase.16.06.02.prd9.SPA.pkg
/flash/cat9k-rpboot.16.06.02.prd9.SPA.pkg
/flash/cat9k-rpbase.16.06.02.prd9.SPA.pkg
/flash/cat9k-guestshell.16.06.02.prd9.SPA.pkg
/flash/cat9k-espbase.16.06.02.prd9.SPA.pkg
/flash/cat9k-cc_srdriver.16.06.02.prd9.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
  [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.16.06.02.prd9.SPA.pkg
      Added cat9k-espbase.16.06.02.prd9.SPA.pkg
      Added cat9k-guestshell.16.06.02.prd9.SPA.pkg
      Added cat9k-rpbase.16.06.02.prd9.SPA.pkg
      Added cat9k-rpboot.16.06.02.prd9.SPA.pkg
      Added cat9k-sipbase.16.06.02.prd9.SPA.pkg
      Added cat9k-sipspa.16.06.02.prd9.SPA.pkg
      Added cat9k-srdriver.16.06.02.prd9.SPA.pkg
      Added cat9k-webui.16.06.02.prd9.SPA.pkg
      Added cat9k-wlc.16.06.02.prd9.SPA.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 20:43:39.249: %IOSXE-5-PLATFORM: Switch 1 R0/0: Oct 30 20:43:39 rollback_timer.sh:

%INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Install auto abort timer will expire in 1800
 seconds
 Install will reload the system now!
SUCCESS: install_activate  Mon Oct 30 20:43:44 UTC 2017

Device#
*Oct 30 20:43:44.615: %IOSXE-5-PLATFORM: Switch 1 R0/0: Oct 30 20:43:44 install_engine.sh:

%INSTALL-5-INSTALL_COMPLETED_INFO: Completed install activate PACKAGE
                                   Chassis 1 reloading, reason - Reload command
```

Verifying Software Install

Procedure

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 Sun Jun 11 15:01:37 Universal 2017
[0|install_op_boot]: END SUCCESS  Sun Jun 11 15:01:44 Universal 2017
[1|install_commit]: START Mon Jun 12 07:27:31 UTC 2017
[1|install_commit(INFO, )]: Releasing transaction lock...
[1|install_commit(CONSOLE, )]: Committing PACKAGE
[remote|install_commit]: START Mon Jun 12 07:28:08 UTC 2017
[remote|install_commit(INFO, )]: Releasing transaction lock...
[remote|install_commit]: END SUCCESS  Mon Jun 12 07:28:41 UTC 2017
[1|install_commit(INFO, )]: [1 2 3]: Performing Commit
  SUCCESS: Commit finished
[1|install_commit(INFO, )]: install_commit: START Mon Jun 12 07:28:08 UTC 2017
  SUCCESS: install_commit  Mon Jun 12 07:28:41 UTC 2017
[1|install_commit(INFO, )]: Remote output from switch 2
[1|install_commit(INFO, )]: install_commit: START Mon Jun 12 07:28:12 UTC 2017
  SUCCESS: install_commit  Mon Jun 12 07:28:44 UTC 2017
[1|install_commit(INFO, )]: install_commit: START Mon Jun 12 07:28:12 UTC 2017
  SUCCESS: install_commit  Mon Jun 12 07:28:45 UTC 2017
[1|install_commit]: END SUCCESS  Mon Jun 12 07:28:47 UTC 2017
```

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

[ 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   C    16.6.1.0

Device#
```

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

Example:

```
Device# show install package flash:cat9k_iosxe.16.06.01.SPA.bin
```

Displays information about the specified software install package file.

```
Device# show install package flash:cat9k_iosxe.16.06.01.SPA.bin

Package: cat9k_iosxe.16.06.01.SPA.bin
  Size: 333806196
  Timestamp: Sun Jun 11 14:47:23 2017 UTC
```

```
  Canonical path: /flash/cat9k_iosxe.16.06.01.SPA.bin

 Raw disk-file SHA1sum:
    5e9ef6ed1f7472b35eddd61df300e44b14b65ec4
  Header size:     1000 bytes
  Package type:    10002
  Package flags:   0
  Header version:  3

  Internal package information:
    Name: cc_srdriver
    BuildTime:
    ReleaseDate: Sun-27-Aug-17-09:05
    BootArchitecture: none
    RouteProcessor: cat9k
    Platform: CAT9K
    User: mcpre
    PackageName: cc_srdriver
    Build: BLD_V166_THROTTLE_LATEST_20170827_090555
    CardTypes:

  Package is not bootable.
Device#
```

Step 5 **show install active**

Example:

```
Device# show install active
```

Displays information about the active software install package.

```
Device# show install active

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

Step 6 **show install inactive**

Example:

```
Device# show install inactive
```

Displays information about the inactive packages.

```
Device# show install inactive

[ R0 ] Inactive Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
IMG   I    16.7.1.0
Device#
```

Step 7 **show install committed**

Example:

```
Device# show install committed
```

Displays information about committed packages.

```
Device# show install committed

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

Step 8 **show install uncommitted**

Example:

```
Device# show install uncommitted
```

Displays information about uncommitted packages.

```
Device# show install uncommitted

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

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]
```

Additional References For Performing Device Setup

Related Documents

Related Topic	Document Title
Device setup commands Boot loader commands	*Command Reference (Catalyst 9500 Series Switches)*
Hardware installation	*Cisco Catalyst 9500 Series Switches Hardware Installation Guide* .

Feature History for Performing Device Setup Configuration

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Everest 16.5.1a	Device Setup Configuration	A device setup configuration can be performed, including auto configuration of IP address assignments and DHCP. Support for this feature was introduced only on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches.
Cisco IOS XE Fuji 16.8.1a	Device Setup Configuration	Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches.
Cisco IOS XE Cupertino 17.7.1	Device Setup Configuration	Support for this feature was introduced on the C9500X-28C8D model of the Cisco Catalyst 9500 Series Switches.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.