



CHAPTER 3

Configuring the Switch for the First Time

This chapter describes how to initially configure a Catalyst 4500 series switch.

The information presented here supplements the administration information and procedures in this publication: *Cisco IOS Configuration Fundamentals Command Reference*, Release 12.2SR, at this URL:

http://www.cisco.com/en/US/docs/ios/12_2/configfun/command/reference/frfabout.html

This chapter includes the following major sections:

- [Default Switch Configuration, page 3-1](#)
- [Configuring DHCP-Based Autoconfiguration, page 3-2](#)
- [Configuring the Switch, page 3-8](#)
- [Controlling Access to Privileged EXEC Commands, page 3-13](#)
- [Recovering a Lost Enable Password, page 3-25](#)
- [Modifying the Supervisor Engine Startup Configuration, page 3-25](#)
- [Resetting a Switch to Factory Default Settings, page 3-32](#)
- [Configuring Crashinfo, page 3-32](#)



Note

For complete syntax and usage information for the switch commands used in this chapter, look at the *Cisco Catalyst 4500 Series Switch Command Reference* and related publications at this location:

<http://www.cisco.com/en/US/products/hw/switches/ps4324/index.html>

If the command is not found in the Catalyst 4500 Command Reference, it is located in the larger Cisco IOS library. Refer to the *Catalyst 4500 Series Switch Cisco IOS Command Reference* and related publications at this location:

<http://www.cisco.com/en/US/products/ps6350/index.html>

Default Switch Configuration

This section describes the default configurations for the Catalyst 4500 series switch. [Table 3-1](#) shows the default configuration settings for each feature.

Table 3-1 Default Switch Configuration

Feature	Default Settings
Administrative connection	Normal mode
Global switch information	No default value for system name, system contact, and location
System clock	No value for system clock time
Passwords	No passwords are configured for normal mode or enable mode (press the Return key)
Switch prompt	Switch>
Interfaces	Enabled, with speed and flow control autonegotiated, and without IP addresses

Configuring DHCP-Based Autoconfiguration

These sections describe how to configure DHCP-based autoconfiguration:

- [DHCP-Based Autoconfiguration, page 3-2](#)
- [DHCP Client Request Process, page 3-3](#)
- [Configuring the DHCP Server, page 3-4](#)
- [Configuring the TFTP Server, page 3-4](#)
- [Configuring the DNS Server, page 3-5](#)
- [Configuring the Relay Device, page 3-5](#)
- [Obtaining Configuration Files, page 3-6](#)
- [Example Configuration, page 3-7](#)

If your DHCP server is a Cisco device, or if you are configuring the switch as a DHCP server, refer to the “IP Addressing and Services” section in the *Cisco IOS IP and IP Routing Configuration Guide for Cisco IOS Release 12.1* for additional information about configuring DHCP.

DHCP-Based Autoconfiguration



Note

Starting with Release 12.2(20)EW, you can enable DHCP AutoConfiguration by entering the **write erase** command. This command clears the startup-config in NVRAM. In images prior to Release 12.2(20)EW, this command will not enable autoconfiguration.

DHCP provides configuration information to Internet hosts and internetworking devices. This protocol consists of two components: one component for delivering configuration parameters from a DHCP server to a device and another component that is a mechanism 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 switch can act as both a DHCP client and a DHCP server.

With DHCP-based autoconfiguration, no DHCP client-side configuration is needed on your switch because your switch (the DHCP client) is automatically configured at startup with IP address information and a configuration file. However, you need to configure the DHCP server or the DHCP server feature on your switch for various lease options associated with IP addresses. If you are using 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.

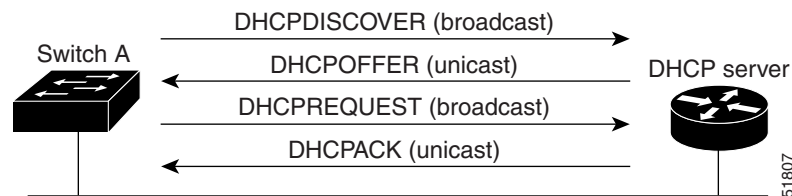
DHCP-based autoconfiguration replaces the BOOTP client functionality on your switch.

DHCP Client Request Process

At startup the switch automatically requests configuration information from a DHCP server if a configuration file is not present on the switch.

Figure 3-1 shows the sequence of messages that are exchanged between the DHCP client and the DHCP server.

Figure 3-1 DHCP Client and Server Message Exchange



The client, Switch 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, 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 the configuration information that it received from the server. The amount of information the switch receives depends on how you configure the DHCP server. For more information, see the “[Configuring the DHCP Server](#)” section on page 3-4.

If the configuration parameters sent to the client in the DHCPOFFER unicast message are invalid (if 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 might have assigned the parameters to another client.)

A DHCP client might receive offers from multiple DHCP servers and can accept any of them; however, the client usually accepts the first offer it receives. The offer from the DHCP server is not a guarantee that the IP address will be allocated to the client; however, the server usually reserves the address until the client has had a chance to formally request the address.

Configuring the DHCP Server

A switch can act as both the DHCP client and the DHCP server. By default, the Cisco IOS DHCP server and relay agent features are enabled on your switch.

You should configure the DHCP server, or the DHCP server feature running on your switch, with reserved leases that are bound to each switch by the switch hardware address.

If you want the switch 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 (required)

**Note**

The router IP address is the default gateway address for the switch.

If you want the switch to receive the configuration file from a TFTP server, you must configure the DHCP server with these lease options:

- TFTP server name or IP address (required)
- Boot filename (the name of the configuration file that the client needs) (recommended)
- Host name (optional)

Depending on the settings of the DHCP server or the DHCP server feature running on your switch, the switch can receive IP address information, the configuration file, or both.

If you do not configure the DHCP server, or the DHCP server feature running on your switch, with the lease options described earlier, the switch replies to client requests with only those parameters that are configured. If the IP address and subnet mask are not in the reply, the switch is not configured. If the router IP address or TFTP server name (or IP address) are not found, the switch might send broadcast, instead of unicast, TFTP requests. Unavailability of other lease options does not impact autoconfiguration.

The DHCP server, or the DHCP server feature running on your switch, can be on the same LAN or on a different LAN than the switch. If the DHCP server is running on a different LAN, you should configure a DHCP relay, which 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. For more information on relay devices, see the [“Configuring the Relay Device” section on page 3-5](#).

Configuring the TFTP Server

Based on the DHCP server configuration, the switch attempts to download one or more configuration files from the TFTP server. If you configured the DHCP server to respond to the switch 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 switch attempts to download the specified configuration file from the specified TFTP server.

If you did not specify the configuration filename or the TFTP server name, or if the configuration file could not be downloaded, the switch attempts to download a configuration file using various combinations of filenames and TFTP server addresses. The files include the specified configuration

filename (if any) and the following files: `network-config`, `cisconet.cfg`, `hostname.config`, or `hostname.cfg`, where *hostname* is the current hostname of the switch and `router-config` and `ciscotr.cfg`. The TFTP server addresses used include the specified TFTP server address (if any) and the broadcast address (255.255.255.255).

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

- The configuration file named in the DHCP reply (the actual switch configuration file).
- The `network-config` or the `cisconet.cfg` file (known as the default configuration files).
- The `router-config` or the `ciscotr.cfg` file. (These files contain commands common to all switches. 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 you plan to use is on a different LAN from the switch, or if you plan to access it with the switch through the broadcast address (which occurs if the DHCP server response does not contain all the required information described earlier), you must configure a relay to forward the TFTP packets to the TFTP server. For more information, see the “[Configuring the Relay Device](#)” section on page 3-5. The preferred solution is to configure either the DHCP server or the DHCP server feature running on your switch with all the required information.

Configuring the DNS Server

The DHCP server, or the DHCP server feature running on your switch, 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 switch.

You can configure the IP addresses of the DNS servers in the lease database of the DHCP server 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 or on a different LAN as the switch. If it is on a different LAN, the switch must be able to access it through a router.

Configuring the Relay Device

You must configure a relay device to forward received broadcast packets to the destination host whenever a switch sends broadcast packets to which a host on a different LAN must respond. Examples of such broadcast packets are DHCP, DNS, and in some cases, TFTP packets.

If the relay device is a Cisco router, enable IP routing (**ip routing** global configuration command), and configure helper addresses (**ip helper-address** interface configuration command). For example, in [Figure 3-2](#), configure the router interfaces as follows:

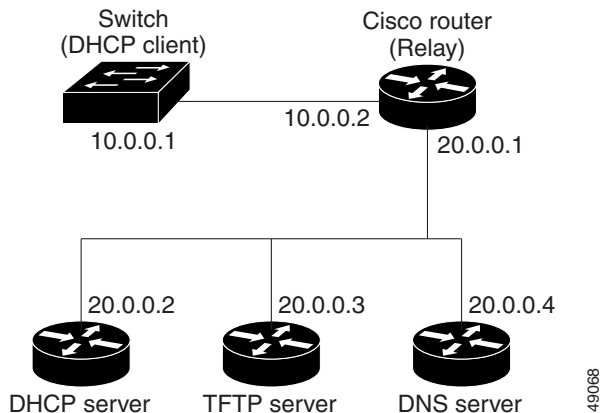
On interface 10.0.0.2:

```
router(config-if)# ip helper-address 20.0.0.2
router(config-if)# ip helper-address 20.0.0.3
router(config-if)# ip helper-address 20.0.0.4
```

On interface 20.0.0.1:

```
router(config-if)# ip helper-address 10.0.0.1
```

Figure 3-2 Relay Device Used in Autoconfiguration



Obtaining Configuration Files

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

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

The switch receives its IP address, subnet mask, TFTP server address, and the configuration filename from either the DHCP server or the DHCP server feature running on your switch. The switch sends a unicast message to the TFTP server to retrieve the named configuration file from the base directory of the server, and upon receipt, completes its boot-up process.

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

The switch receives its IP address, subnet mask, and the configuration filename from either the DHCP server or the DHCP server feature running on your switch. The switch sends a broadcast message to a TFTP server to retrieve the named configuration file from the base directory of the server, and upon receipt, completes its boot-up process.

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

The switch receives its IP address, subnet mask, and the TFTP server address from either the DHCP server or the DHCP server feature running on your switch. The switch sends a unicast message to the TFTP server to retrieve the network-config or ciscoconet.cfg default configuration file. (If the network-config file cannot be read, the switch reads the ciscoconet.cfg file.)

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

After obtaining its host name from the default configuration file or the DHCP reply, the switch reads the configuration file that has the same name as its host name (*hostname-config* or *hostname.cfg*, depending on whether or not the network-config file or the ciscoconet.cfg file was read earlier) from the TFTP server. If the ciscoconet.cfg file is read, the filename of the host is truncated to eight characters.

If the switch cannot read the network-config, cisco.net.cfg, or the hostname file, it reads the router-config file. If the switch cannot read the router-config file, it reads the ciscotr.cfg file.



Note

The switch broadcasts TFTP server requests provided that one of these conditions is met: the TFTP server is not obtained from the DHCP replies; all attempts to read the configuration file through unicast transmissions fail; or the TFTP server name cannot be resolved to an IP address.

Example Configuration

Figure 3-3 shows a network example for retrieving IP information using DHCP-based autoconfiguration.

Figure 3-3 DHCP-Based Autoconfiguration Network Example

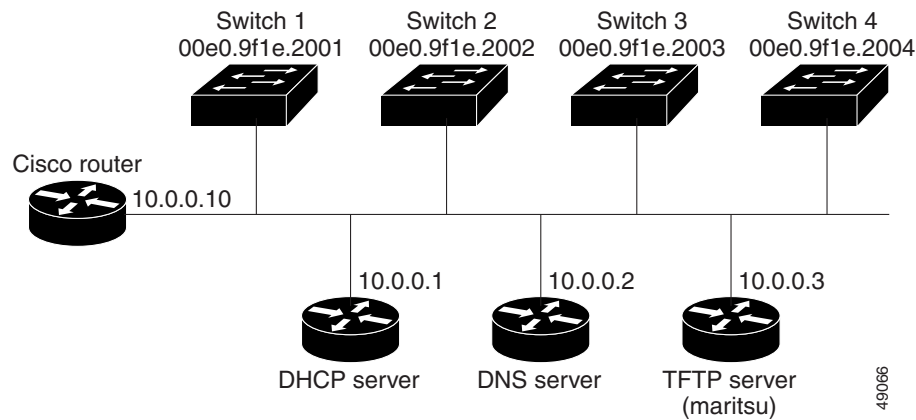


Table 3-2 shows the configuration of the reserved leases on either the DHCP server or the DHCP server feature running on your switch.

Table 3-2 DHCP Server Configuration

	Switch 1	Switch 2	Switch 3	Switch 4
Binding key (hardware address)	00e0.9f1e.2001	00e0.9f1e.2002	00e0.9f1e.2003	00e0.9f1e.2004
IP address	10.0.0.21	10.0.0.22	10.0.0.23	10.0.0.24
Subnet mask	255.255.255.0	255.255.255.0	255.255.255.0	255.255.255.0
Router address	10.0.0.10	10.0.0.10	10.0.0.10	10.0.0.10
DNS server address	10.0.0.2	10.0.0.2	10.0.0.2	10.0.0.2
TFTP server name	maritsu or 10.0.0.3	maritsu or 10.0.0.3	maritsu or 10.0.0.3	maritsu or 10.0.0.3
Boot filename (configuration file) (optional)	switch1-config	switch2-config	switch3-config	switch4-config
Host name (optional)	switch1	switch2	switch3	switch4

DNS Server Configuration

The DNS server maps the TFTP server name *maritsu* to IP address 10.0.0.3.

TFTP Server Configuration (on UNIX)

The TFTP server base directory is set to `/tftpserver/work/`. This directory contains the `network-confg` file used in the two-file read method. This file contains the host name that you plan to assign to the switch based on its IP address. The base directory also contains a configuration file for each switch (*switch1-confg*, *switch2-confg*, and so forth) as shown in the following display:

```
prompt> cd /tftpserver/work/
prompt> ls
network-confg
switch1-confg
switch2-confg
switch3-confg
switch4-confg
prompt> cat network-confg
ip host switch1 10.0.0.21
ip host switch2 10.0.0.22
ip host switch3 10.0.0.23
ip host switch4 10.0.0.24
```

DHCP Client Configuration

No configuration file is present on Switch 1 through Switch 4.

Configuration Explanation

In [Figure 3-3](#), Switch 1 reads its configuration file as follows:

- Switch 1 obtains its IP address 10.0.0.21 from the DHCP server.
- If no configuration filename is given in the DHCP server reply, Switch 1 reads the `network-confg` file from the base directory of the TFTP server.
- Switch 1 adds the contents of the `network-confg` file to its host table.
- Switch 1 reads its host table by indexing its IP address 10.0.0.21 to its host name (`switch1`).
- Switch 1 reads the configuration file that corresponds to its host name; for example, it reads *switch1-confg* from the TFTP server.

Switches 2 through 4 retrieve their configuration files and IP addresses in the same way.

Configuring the Switch

The following sections describe how to configure your switch:

- [Using Configuration Mode to Configure Your Switch, page 3-9](#)
- [Verifying the Running Configuration Settings, page 3-9](#)
- [Saving the Running Configuration Settings to Your Start-Up File, page 3-10](#)
- [Reviewing the Configuration in NVRAM, page 3-10](#)
- [Configuring a Default Gateway, page 3-11](#)
- [Configuring a Static Route, page 3-11](#)

Using Configuration Mode to Configure Your Switch

To configure your switch from configuration mode, perform this procedure:

- Step 1** Connect a console terminal to the console interface of your supervisor engine.
- Step 2** After a few seconds, you will see the user EXEC prompt (**Switch>**). Now, you may want to enter privileged EXEC mode, also known as enable mode. Type **enable** to enter enable mode:

```
Switch> enable
```



Note You must be in enable mode to make configuration changes.

The prompt will change to the enable prompt (#):

```
Switch#
```

- Step 3** At the enable prompt (#), enter the **configure terminal** command to enter global configuration mode:

```
Switch# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Switch(config)#
```

- Step 4** At the global configuration mode prompt, enter the **interface type slot/interface** command to enter interface configuration mode:

```
Switch(config)# interface fastethernet 5/1
Switch(config-if)#
```

- Step 5** In either of these configuration modes, enter changes to the switch configuration.
- Step 6** Enter the **end** command to exit configuration mode.
- Step 7** Save your settings. (See the [“Saving the Running Configuration Settings to Your Start-Up File”](#) section on page 3-10.)

Your switch is now minimally configured and can boot with the configuration you entered. To see a list of the configuration commands, enter ? at the prompt or press the **help** key in configuration mode.

Verifying the Running Configuration Settings

To verify the configuration settings you entered or the changes you made, enter the **show running-config** command at the enable prompt (#), as shown in this example:

```
Switch# show running-config
Building configuration...

Current configuration:
!
version 12.0
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname Switch
```

```

<...output truncated...>

!
line con 0
  transport input none
line vty 0 4
  exec-timeout 0 0
  password lab
  login
  transport input lat pad dsipcon mop telnet rlogin udptn nasi
!
end
Switch#

```

Saving the Running Configuration Settings to Your Start-Up File



Caution

This command saves the configuration settings that you created in configuration mode. If you fail to do this step, your configuration will be lost the next time you reload the system.

To store the configuration, changes to the configuration, or changes to the startup configuration in NVRAM, enter the **copy running-config startup-config** command at the enable prompt (#), as follows:

```
Switch# copy running-config startup-config
```

Reviewing the Configuration in NVRAM

To display information stored in NVRAM, enter the **show startup-config EXEC** command.

The following example shows a typical system configuration:

```

Switch# show startup-config
Using 1579 out of 491500 bytes, uncompressed size = 7372 bytes
Uncompressed configuration from 1579 bytes to 7372 bytes
!
version 12.1
no service pad
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
service compress-config
!
hostname Switch
!
!
ip subnet-zero
!
!
!
interface GigabitEthernet1/1
  no snmp trap link-status
!
interface GigabitEthernet1/2
  no snmp trap link-status
!--More--

<...output truncated...>

```

```

!
line con 0
  exec-timeout 0 0
  transport input none
line vty 0 4
  exec-timeout 0 0
  password lab
  login
  transport input lat pad dsipcon mop telnet rlogin udptn nasi
!
end

Switch#

```

Configuring a Default Gateway



Note

The switch uses the default gateway only when it is not configured with a routing protocol.

Configure a default gateway to send data to subnets other than its own when the switch is not configured with a routing protocol. The default gateway must be the IP address of an interface on a router that is directly connected to the switch.

To configure a default gateway, perform this task:

	Command	Purpose
Step 1	Switch(config)# ip default-gateway <i>IP-address</i>	Configures a default gateway.
Step 2	Switch# show ip route	Verifies that the default gateway is correctly displayed in the IP routing table.

This example shows how to configure a default gateway and how to verify the configuration:

```

Switch# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Switch(config)# ip default-gateway 172.20.52.35
Switch(config)# end
3d17h: %SYS-5-CONFIG_I: Configured from console by console
Switch# show ip route
Default gateway is 172.20.52.35

Host                Gateway                Last Use    Total Uses  Interface
ICMP redirect cache is empty
Switch#

```

Configuring a Static Route

If your Telnet station or SNMP network management workstation is on a different network from your switch and a routing protocol has not been configured, you might need to add a static routing table entry for the network where your end station is located.

To configure a static route, perform this task:

	Command	Purpose
Step 1	Switch(config)# ip route <i>dest_IP_address mask</i> { <i>forwarding_IP</i> vlan <i>vlan_ID</i> }	Configures a static route to the remote network.
Step 2	Switch# show running-config	Verifies that the static route is displayed correctly.

This example shows how to use the **ip route** command to configure a static route to a workstation at IP address 171.10.5.10 on the switch with a subnet mask and IP address 172.20.3.35 of the forwarding router:

```
Switch# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Switch(config)# ip route 171.10.5.10 255.255.255.255 172.20.3.35
Switch(config)# end
Switch#
```

This example shows how to use the **show running-config** command to confirm the configuration of the static route:

```
Switch# show running-config
Building configuration...
.
<...output truncated...>
.
ip default-gateway 172.20.52.35
ip classless
ip route 171.10.5.10 255.255.255.255 172.20.3.35
no ip http server
!
line con 0
  transport input none
line vty 0 4
  exec-timeout 0 0
  password lab
  login
  transport input lat pad dsipcon mop telnet rlogin udptn nasi
!
end

Switch#
```

This example shows how to use the **ip route** command to configure the static route IP address 171.20.5.3 with subnet mask and connected over VLAN 1 to a workstation on the switch:

```
Switch# configure terminal
Switch(config)# ip route 171.20.5.3 255.255.255.255 vlan 1
Switch(config)# end
Switch#
```

This example shows how to use the **show running-config** command to confirm the configuration of the static route:

```
Switch# show running-config
Building configuration...
.
<...output truncated...>
.
```

```

ip default-gateway 172.20.52.35
ip classless
ip route 171.20.5.3 255.255.255.255 Vlan1
no ip http server
!
!
x25 host z
!
line con 0
  transport input none
line vty 0 4
  exec-timeout 0 0
  password lab
  login
  transport input lat pad dsipcon mop telnet rlogin udptn nasi
!
end

Switch#

```

Controlling Access to Privileged EXEC Commands

The procedures in these sections let you control access to the system configuration file and privileged EXEC commands:

- [Setting or Changing a Static enable Password, page 3-13](#)
- [Using the enable password and enable secret Commands, page 3-14](#)
- [Setting or Changing a Privileged Password, page 3-14](#)
- [Encrypting Passwords, page 3-22](#)
- [Encrypting Passwords, page 3-22](#)
- [Configuring Multiple Privilege Levels, page 3-23](#)

Setting or Changing a Static enable Password

To set or change a static password that controls access to the enable mode, perform this task:

Command	Purpose
Switch(config)# enable password <i>password</i>	Sets a new password or changes an existing password for the privileged EXEC mode.

This example shows how to configure an enable password as “lab” at the privileged EXEC mode:

```

Switch# configure terminal
Switch(config)# enable password lab
Switch(config)#

```

For instructions on how to display the password or access level configuration, see the “[Displaying the Password, Access Level, and Privilege Level Configuration](#)” section on page 3-24.

Using the enable password and enable secret Commands

To provide an additional layer of security, particularly for passwords that cross the network or that are stored on a TFTP server, you can use either the **enable password** or **enable secret** command. Both commands configure an encrypted password that you must enter to access the enable mode (the default) or any other privilege level that you specify.

We recommend that you use the **enable secret** command.

If you configure the **enable secret** command, it takes precedence over the **enable password** command; the two commands cannot be in effect simultaneously.

To configure the switch to require an enable password, perform either one of these tasks:

Command	Purpose
Switch(config)# enable password [level level] { <i>password</i> <i>encryption-type encrypted-password</i> }	Establishes a password for the privileged EXEC mode.
Switch(config)# enable secret [level level] { <i>password</i> <i>encryption-type encrypted-password</i> }	Specifies a secret password that will be saved using a nonreversible encryption method. (If enable password and enable secret commands are both set, users must enter the enable secret password.)

When you enter either of these password commands with the **level** option, you define a password for a specific privilege level. After you specify the level and set a password, give the password only to users who need to have access at this level. Use the **privilege level** configuration command to specify commands accessible at various levels.

If you enable the **service password-encryption** command, the password you enter is encrypted. When you display the password with the **more system:running-config** command, the password displays the password in encrypted form.

If you specify an encryption type, you must provide an encrypted password—an encrypted password you copy from another Catalyst 4500 series switch configuration.



Note

You cannot recover a lost encrypted password. You must clear NVRAM and set a new password. See the [“Recovering a Lost Enable Password”](#) section on page 3-25 for more information.

For information on how to display the password or access level configuration, see the [“Displaying the Password, Access Level, and Privilege Level Configuration”](#) section on page 3-24.

Setting or Changing a Privileged Password

To set or change a privileged password, perform this task:

Command	Purpose
Switch(config-line)# password <i>password</i>	Sets a new password or changes an existing password for the privileged level.

For information on how to display the password or access level configuration, see the “[Displaying the Password, Access Level, and Privilege Level Configuration](#)” section on page 3-24.

Controlling Switch Access with TACACS+

This section describes how to enable and configure TACACS+, which provides detailed accounting information and flexible administrative control over authentication and authorization processes. TACACS+ is facilitated through authentication, authorization, accounting (AAA) and can be enabled only through AAA commands.

**Note**

For complete syntax and usage information for the commands used in this section, see the *Cisco IOS Security Command Reference, Release 12.2*.

This section contains this configuration information:

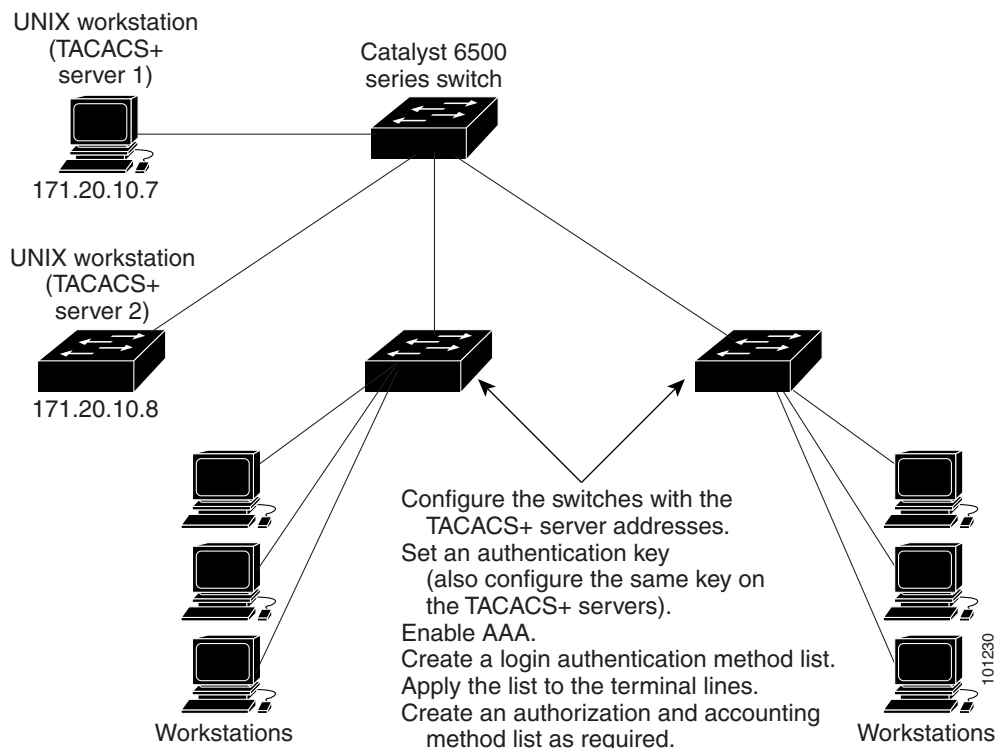
- [Understanding TACACS+, page 3-15](#)
- [TACACS+ Operation, page 3-17](#)
- [Configuring TACACS+, page 3-17](#)
- [Displaying the TACACS+ Configuration, page 3-22](#)

Understanding TACACS+

TACACS+ is a security application that provides centralized validation of users attempting to gain access to your switch. TACACS+ services are maintained in a database on a TACACS+ daemon typically running on a UNIX or Windows NT workstation. You should have access to and should configure a TACACS+ server before configuring TACACS+ features on your switch.

TACACS+ provides for separate and modular AAA facilities. TACACS+ allows for a single access control server (the TACACS+ daemon) to provide each service—authentication, authorization, and accounting—independently. Each service can be locked into its own database to take advantage of other services available on that server or on the network, depending on the capabilities of the daemon.

The goal of TACACS+ is to provide a method for managing multiple network access points from a single management service. Your switch can be a network access server along with other Cisco routers and access servers. A network access server provides connections to a single user, to a network or subnetwork, and to interconnected networks as shown in [Figure 3-4](#).

Figure 3-4 Typical TACACS+ Network Configuration

TACACS+ administered through the AAA security services can provide these services:

- **Authentication**—Provides complete control of authentication through login and password dialog, challenge and response, and messaging support.
The authentication facility can conduct a dialog with the user (such as, after a username and password are provided, to challenge a user with several questions such as home address, mother's maiden name, service type, and social security number). The TACACS+ authentication service can also send messages to user screens. For example, a message could notify users that their passwords must be changed because of the company's password aging policy.
- **Authorization**—Provides strict control over user capabilities for the duration of the user's session, including but not limited to setting autocommands, access control, session duration, or protocol support. You can also enforce restrictions on the commands a user can execute with the TACACS+ authorization feature.
- **Accounting**—Collects and sends information used for billing, auditing, and reporting to the TACACS+ daemon. Network managers can use the accounting facility to track user activity for a security audit or to provide information for user billing. Accounting records include user identities, start and stop times, executed commands (such as PPP), number of packets, and number of bytes.

The TACACS+ protocol provides authentication between the switch and the TACACS+ daemon, and it ensures confidentiality because all protocol exchanges between the switch and the TACACS+ daemon are encrypted.

You need a system running the TACACS+ daemon software to use TACACS+ on your switch.

TACACS+ Operation

When a user attempts a simple ASCII login by authenticating to a switch using TACACS+, this process occurs:

1. When the connection is established, the switch contacts the TACACS+ daemon to obtain a username prompt, which is then displayed to the user. The user enters a username, and the switch then contacts the TACACS+ daemon to obtain a password prompt. The switch displays the password prompt to the user, the user enters a password, and the password is then sent to the TACACS+ daemon.

TACACS+ allows a conversation between the daemon and the user until the daemon receives enough information to authenticate the user. The daemon prompts for a username and password combination, but can include other items such as the user's mother's maiden name.

2. The switch eventually receives one of these responses from the TACACS+ daemon:
 - **ACCEPT**—The user is authenticated and service can begin. If the switch is configured to require authorization, authorization begins at this time.
 - **REJECT**—The user is not authenticated. The user can be denied access or is prompted to retry the login sequence, depending on the TACACS+ daemon.
 - **ERROR**—An error occurred at some time during authentication with the daemon or in the network connection between the daemon and the switch. If an **ERROR** response is received, the switch typically tries to use an alternative method for authenticating the user.
 - **CONTINUE**—The user is prompted for additional authentication information.

After authentication, the user undergoes an additional authorization phase if authorization has been enabled on the switch. Users must first successfully complete TACACS+ authentication before proceeding to TACACS+ authorization.

3. If TACACS+ authorization is required, the TACACS+ daemon is again contacted, and it returns an **ACCEPT** or **REJECT** authorization response. If an **ACCEPT** response is returned, the response contains data in the form of attributes that direct the **EXEC** or **NETWORK** session for that user and the services that the user can access:
 - Telnet, Secure Shell (SSH), rlogin, or privileged **EXEC** services
 - Connection parameters, including the host or client IP address, access list, and user timeouts

Configuring TACACS+

This section describes how to configure your switch to support TACACS+. At a minimum, you must identify the host or hosts maintaining the TACACS+ daemon and define the method lists for TACACS+ authentication. You can optionally define method lists for TACACS+ authorization and accounting. A method list defines the sequence and methods used to authenticate, to authorize, or to keep accounts on a user. You can use method lists to designate one or more security protocols, ensuring a backup system if the initial method fails. The software uses the first method listed to authenticate, to authorize, or to keep accounts on users; if that method does not respond, the software selects the next method in the list. This process continues until there is successful communication with a listed method or the method list is exhausted.

This section contains this configuration information:

- [Default TACACS+ Configuration, page 3-18](#)
- [Identifying the TACACS+ Server Host and Setting the Authentication Key, page 3-18](#)
- [Configuring TACACS+ Login Authentication, page 3-19](#)
- [Configuring TACACS+ Authorization for Privileged EXEC Access and Network Services, page 3-21](#)
- [Starting TACACS+ Accounting, page 3-21](#)

Default TACACS+ Configuration

TACACS+ and AAA are disabled by default.

To prevent a lapse in security, you cannot configure TACACS+ through a network management application. When enabled, TACACS+ can authenticate users accessing the switch through the CLI.



Note

Although TACACS+ configuration is performed through the CLI, the TACACS+ server authenticates HTTP connections that have been configured with a privilege level of 15.

Identifying the TACACS+ Server Host and Setting the Authentication Key

You can configure the switch to use a single server or AAA server groups in order to group existing server hosts for authentication. You can group servers to select a subset of the configured server hosts and use them for a particular service. The server group is used with a global server-host list and contains the list of IP addresses of the selected server hosts.

Beginning in privileged EXEC mode, follow these steps to identify the IP host or host maintaining TACACS+ server and optionally set the encryption key:

	Command	Purpose
Step 1	<code>configure terminal</code>	Enters global configuration mode.
Step 2	<code>tacacs-server host hostname [port integer] [timeout integer] [key string]</code>	Identifies the IP host or hosts maintaining a TACACS+ server. Enter this command multiple times to create a list of preferred hosts. The software searches for hosts in the order in which you specify them. <ul style="list-style-type: none"> • For <i>hostname</i>, specify the name or IP address of the host. • (Optional) For port integer, specify a server port number. The default is port 49. The range is 1 to 65535. • (Optional) For timeout integer, specify a time in seconds the switch waits for a response from the daemon before it times out and declares an error. The default is 5 seconds. The range is 1 to 1000 seconds. • (Optional) For key string, specify the encryption key for encrypting and decrypting all traffic between the switch and the TACACS+ daemon. You must configure the same key on the TACACS+ daemon for encryption to succeed.
Step 3	<code>aaa new-model</code>	Enables AAA.
Step 4	<code>aaa group server tacacs+ group-name</code>	(Optional) Defines the AAA server-group with a group name. This command puts the switch in a server group subconfiguration mode.

	Command	Purpose
Step 5	<code>server ip-address</code>	(Optional) Associates a particular TACACS+ server with the defined server group. Repeat this step for each TACACS+ server in the AAA server group. Each server in the group must be previously defined in Step 2.
Step 6	<code>end</code>	Returns to privileged EXEC mode.
Step 7	<code>show tacacs</code>	Verifies your entries.
Step 8	<code>copy running-config startup-config</code>	(Optional) Saves your entries in the configuration file.

To remove the specified TACACS+ server name or address, use the **no tacacs-server host *hostname*** global configuration command. To remove a server group from the configuration list, use the **no aaa group server tacacs+ *group-name*** global configuration command. To remove the IP address of a TACACS+ server, use the **no server ip-address** server group subconfiguration command.

Configuring TACACS+ Login Authentication

To configure AAA authentication, define a named list of authentication methods and then apply that list to various ports. The method list defines the types of authentication you intend to perform and the sequence in which you intend to perform them; you must apply the list to a specific port before you can perform any of the defined authentication methods. The only exception is the default method list (which, by coincidence, is named *default*). The default method list is automatically applied to all ports except those that have a named method list explicitly defined. A defined method list overrides the default method list.

A method list describes the sequence and authentication methods that must be queried to authenticate a user. You can designate one or more security protocols for authentication, ensuring a backup system for authentication in case the initial method fails. The software uses the first method listed to authenticate users; if that method fails to respond, the software selects the next authentication method in the method list. This process continues until there is successful communication with a listed authentication method or until all defined methods are exhausted. If authentication fails at any point in this cycle—meaning that the security server or local username database responds by denying the user access—the authentication process stops, and no other authentication methods are attempted.

Beginning in privileged EXEC mode, follow these steps to configure login authentication:

	Command	Purpose
Step 1	<code>configure terminal</code>	Enters global configuration mode.
Step 2	<code>aaa new-model</code>	Enables AAA.

	Command	Purpose
Step 3	<code>aaa authentication login {default list-name} method1 [method2...]</code>	<p>Creates a login authentication method list.</p> <ul style="list-style-type: none"> To create a default list that is used when a named list is <i>not</i> specified in the login authentication command, use the default keyword followed by the methods that you plan to use in default situations. The default method list is automatically applied to all ports. For <i>list-name</i>, specify a character string to name the list you are creating. For <i>method1...</i>, specify the actual method the authentication algorithm tries. The additional methods of authentication are used only if the previous method returns an error, not if it fails. <p>Select one of these methods:</p> <ul style="list-style-type: none"> enable—Use the enable password for authentication. Before you can use this authentication method, you must define an enable password by using the enable password global configuration command. group tacacs+—Uses TACACS+ authentication. Before you can use this authentication method, you must configure the TACACS+ server. For more information, see the “Identifying the TACACS+ Server Host and Setting the Authentication Key” section on page 3-18. line—Use the line password for authentication. Before you can use this authentication method, you must define a line password. Use the password password line configuration command. local—Use the local username database for authentication. You must enter username information in the database. Use the username password global configuration command. local-case—Use a case-sensitive local username database for authentication. You must enter username information in the database by using the username name password global configuration command. none—Do not use any authentication for login.
Step 4	<code>line [console tty vty] line-number [ending-line-number]</code>	Enters line configuration mode, and configure the lines to which you want to apply the authentication list.
Step 5	<code>login authentication {default list-name}</code>	<p>Applies the authentication list to a line or set of lines.</p> <ul style="list-style-type: none"> If you specify default, use the default list created with the aaa authentication login command. For <i>list-name</i>, specify the list created with the aaa authentication login command.
Step 6	<code>end</code>	Returns to privileged EXEC mode.
Step 7	<code>show running-config</code>	Verifies your entries.
Step 8	<code>copy running-config startup-config</code>	(Optional) Saves your entries in the configuration file.

To disable AAA, use the **no aaa new-model** global configuration command. To disable AAA authentication, use the **no aaa authentication login {default | list-name} method1 [method2...]** global configuration command. To either disable TACACS+ authentication for logins or to return to the default value, use the **no login authentication {default | list-name}** line configuration command.

Configuring TACACS+ Authorization for Privileged EXEC Access and Network Services

AAA authorization limits the services available to a user. When AAA authorization is enabled, the switch uses information retrieved from the user's profile, which is located either in the local user database or on the security server, to configure the user's session. The user is granted access to a requested service only if the information in the user profile allows it.

You can use the **aaa authorization** global configuration command with the **tacacs+** keyword to set parameters that restrict a user's network access to privileged EXEC mode.

The **aaa authorization exec tacacs+ local** command sets these authorization parameters:

- Use TACACS+ for privileged EXEC access authorization if authentication was performed by using TACACS+.
- Use the local database if authentication was not performed by using TACACS+.



Note

Authorization is bypassed for authenticated users who log in through the CLI even if authorization has been configured.

Beginning in privileged EXEC mode, follow these steps to specify TACACS+ authorization for privileged EXEC access and network services:

	Command	Purpose
Step 1	<code>configure terminal</code>	Enters global configuration mode.
Step 2	<code>aaa authorization network tacacs+</code>	Configures the switch for user TACACS+ authorization for all network-related service requests.
Step 3	<code>aaa authorization exec tacacs+</code>	Configures the switch for user TACACS+ authorization if the user has privileged EXEC access. The exec keyword might return user profile information (such as autocommand information).
Step 4	<code>end</code>	Returns to privileged EXEC mode.
Step 5	<code>show running-config</code>	Verifies your entries.
Step 6	<code>copy running-config startup-config</code>	(Optional) Saves your entries in the configuration file.

To disable authorization, use the **no aaa authorization {network | exec} method1** global configuration command.

Starting TACACS+ Accounting

The AAA accounting feature tracks the services that users are accessing and the amount of network resources that they are consuming. When AAA accounting is enabled, the switch reports user activity to the TACACS+ security server in the form of accounting records. Each accounting record contains accounting attribute-value (AV) pairs and is stored on the security server. This data can then be analyzed for network management, client billing, or auditing.

Beginning in privileged EXEC mode, follow these steps to enable TACACS+ accounting for each Cisco IOS privilege level and for network services:

	Command	Purpose
Step 1	<code>configure terminal</code>	Enters global configuration mode.
Step 2	<code>aaa accounting network start-stop tacacs+</code>	Enables TACACS+ accounting for all network-related service requests.
Step 3	<code>aaa accounting exec start-stop tacacs+</code>	Enables TACACS+ accounting to send a start-record accounting notice at the beginning of a privileged EXEC process and a stop-record at the end.
Step 4	<code>end</code>	Returns to privileged EXEC mode.
Step 5	<code>show running-config</code>	Verifies your entries.
Step 6	<code>copy running-config startup-config</code>	(Optional) Saves your entries in the configuration file.

To disable accounting, use the `no aaa accounting {network | exec} {start-stop} method1...` global configuration command.

Displaying the TACACS+ Configuration

To display TACACS+ server statistics, use the `show tacacs` privileged EXEC command.

Encrypting Passwords

Because protocol analyzers can examine packets (and read passwords), you can increase access security by configuring the Cisco IOS software to encrypt passwords. Encryption prevents the password from being readable in the configuration file.

To configure the Cisco IOS software to encrypt passwords, perform this task:

Command	Purpose
Switch(config)# <code>service password-encryption</code>	Encrypts a password.

Encryption occurs when the current configuration is written or when a password is configured. Password encryption is applied to all passwords, including authentication key passwords, the privileged command password, console and virtual terminal line access passwords, and Border Gateway Protocol (BGP) neighbor passwords. The `service password-encryption` command keeps unauthorized individuals from viewing your password in your configuration file.



Caution

The `service password-encryption` command does not provide a high level of network security. If you use this command, you should also take additional network security measures.

Although you cannot recover a lost encrypted password (that is, you cannot get the original password back), you can regain control of the switch after having lost or forgotten the encrypted password. See the [“Recovering a Lost Enable Password”](#) section on page 3-25 for more information.

For information on how to display the password or access level configuration, see the [“Displaying the Password, Access Level, and Privilege Level Configuration”](#) section on page 3-24.

Configuring Multiple Privilege Levels

By default, Cisco IOS software has two modes of password security: user EXEC mode and privileged EXEC mode. You can configure up to 16 hierarchical levels of commands for each mode. By configuring multiple passwords, you can allow different sets of users to have access to specified commands.

For example, if you want many users to have access to the **clear line** command, you can assign it level 2 security and distribute the level 2 password fairly widely. If you want more restricted access to the **configure** command, you can assign it level 3 security and distribute that password to fewer users.

The procedures in the following sections describe how to configure additional levels of security:

- [Setting the Privilege Level for a Command, page 3-23](#)
- [Changing the Default Privilege Level for Lines, page 3-23](#)
- [Logging In to a Privilege Level, page 3-24](#)
- [Exiting a Privilege Level, page 3-24](#)
- [Displaying the Password, Access Level, and Privilege Level Configuration, page 3-24](#)

Setting the Privilege Level for a Command

To set the privilege level for a command, perform this task:

	Command	Purpose
Step 1	Switch(config)# privilege mode level level <i>command</i>	Sets the privilege level for a command.
Step 2	Switch(config)# enable password level level <i>[encryption-type] password</i>	Specifies the enable password for a privilege level.

For information on how to display the password or access level configuration, see the [“Displaying the Password, Access Level, and Privilege Level Configuration”](#) section on page 3-24.

Changing the Default Privilege Level for Lines

To change the default privilege level for a given line or a group of lines, perform this task:

Command	Purpose
Switch(config-line)# privilege level level	Changes the default privilege level for the line.

For information on how to display the password or access level configuration, see the [“Displaying the Password, Access Level, and Privilege Level Configuration”](#) section on page 3-24.

Logging In to a Privilege Level

To log in at a specified privilege level, perform this task:

Command	Purpose
Switch# enable <i>level</i>	Logs in to a specified privilege level.

Exiting a Privilege Level

To exit to a specified privilege level, perform this task:

Command	Purpose
Switch# disable <i>level</i>	Exits to a specified privilege level.

Displaying the Password, Access Level, and Privilege Level Configuration

To display detailed password information, perform this task:

	Command	Purpose
Step 1	Switch# show running-config	Displays the password and access level configuration.
Step 2	Switch# show privilege	Shows the privilege level configuration.

This example shows how to display the password and access level configuration:

```
Switch# show running-config
Building configuration...

Current configuration:
!
version 12.0
service timestamps debug datetime localtime
service timestamps log datetime localtime
no service password-encryption
!
hostname Switch
!
boot system flash sup-bootflash
enable password lab
!
<...output truncated...>
```

This example shows how to display the privilege level configuration:

```
Switch# show privilege
Current privilege level is 15
Switch#
```


Recovering a Lost Enable Password

**Note**

For more information on the configuration register which is preconfigured in NVRAM, see [“Configuring the Software Configuration Register”](#) section on page 3-26.

To recover a lost enable password, follow these steps:

-
- Step 1** Connect to the console interface.
 - Step 2** Stop the boot sequence and enter ROM monitor by pressing **Ctrl-C** during the first 5 seconds of bootup.
 - Step 3** Configure the switch to boot-up without reading the configuration memory (NVRAM).
 - Step 4** Reboot the system.
 - Step 5** Access enable mode (this can be done without a password if a password has not been configured).
 - Step 6** View or change the password, or erase the configuration.
 - Step 7** Reconfigure the switch to boot-up and read the NVRAM as it normally does.
 - Step 8** Reboot the system.
-

Modifying the Supervisor Engine Startup Configuration

These sections describe how the startup configuration on the supervisor engine works and how to modify the BOOT variable and the configuration register:

- [Understanding the Supervisor Engine Boot Configuration, page 3-25](#)
- [Configuring the Software Configuration Register, page 3-26](#)
- [Specifying the Startup System Image, page 3-30](#)
- [Controlling Environment Variables, page 3-31](#)

Understanding the Supervisor Engine Boot Configuration

The supervisor engine boot process involves two software images: ROM monitor and supervisor engine software. When the switch is booted or reset, the ROMMON code is executed. Depending on the NVRAM configuration, the supervisor engine either stays in ROMMON mode or loads the supervisor engine software.

Two user-configurable parameters determine how the switch boots: the configuration register and the BOOT environment variable. The configuration register is described in the [“Modifying the Boot Field and Using the boot Command”](#) section on page 3-27. The BOOT environment variable is described in the [“Specifying the Startup System Image”](#) section on page 3-30.

Understanding the ROM Monitor

The ROM monitor (ROMMON) is invoked at switch bootup, reset, or when a fatal exception occurs. The switch enters ROMMON mode if the switch does not find a valid software image, if the NVRAM configuration is corrupted, or if the configuration register is set to enter ROMMON mode. From ROMMON mode, you can manually load a software image from bootflash or a flash disk, or you can boot up from the management interface. ROMMON mode loads a primary image from which you can configure a secondary image to boot up from a specified source either locally or through the network using the BOOTLDR environment variable. This variable is described in the “[Switch#](#)” section on [page 3-32](#).

You can also enter ROMMON mode by restarting the switch and then pressing **Ctrl-C** during the first five seconds of startup.



Note

Ctrl-C is always enabled for five seconds after you reboot the switch, regardless of whether the configuration-register setting has **Ctrl-C** disabled.

The ROM monitor has these features:

- Power-on confidence test
- Hardware initialization
- Boot capability (manual bootup and autoboot)
- File system (read-only while in ROMMON)

Configuring the Software Configuration Register

The switch uses a 16-bit software configuration register, which allows you to set specific system parameters. Settings for the software configuration register are preconfigured in NVRAM.

Here are some reasons why you might want to change the software configuration register settings:

- To select a boot source and default boot filename
- To control broadcast addresses
- To set the console terminal baud rate
- To load operating software from flash memory
- To recover a lost password
- To manually boot the system using the **boot** command at the bootstrap program prompt
- To force an automatic bootup from the system bootstrap software (boot image) or from a default system image in onboard flash memory, and read any **boot system** commands that are stored in the configuration file in NVRAM



Caution

To avoid possibly halting the Catalyst 4500 series switch, remember that valid configuration register settings might be combinations of settings and not just the individual settings listed in [Table 3-3](#). For example, the factory default value of 0x2101 is a combination of settings.

[Table 3-3](#) lists the meaning of each of the software configuration memory bits. [Table 3-4](#) defines the *boot field*.

Table 3-3 Software Configuration Register Bits

Bit Number ¹	Hexadecimal	Meaning
00 to 03	0x0000 to 0x000F	Boot field (see Table 3-4)
04	0x0010	Unused
05	0x0020	Bit two of console line speed
06	0x0040	Causes system software to ignore NVRAM contents
07	0x0080	OEM ² bit enabled
08	0x0100	Unused
09	0x0200	Unused
10	0x0400	IP broadcast with all zeros
11 to 12	0x0800 to 0x1000	Bits one and zero of Console line speed (default is 9600 baud)
13	0x2000	Loads ROM monitor after netboot fails
14	0x4000	IP broadcasts do not have network numbers

1. The factory default value for the configuration register is 0x2101. This value is a combination of the following: binary bit 13, bit 8 = 0x0100 and binary bits 00 through 03 = 0x0001. (See [Table 3-4](#).)
2. OEM = original equipment manufacturer.

Table 3-4 Explanation of Boot Field (Configuration Register Bits 00 to 03)

Boot Field	Meaning
00	Stays at the system bootstrap prompt (does not autoboot).
01	Boots the first file in onboard flash memory.
02 to 0F	Autoboots using image(s) specified by the BOOT environment variable. If more than one image is specified, the switch attempts to boot the first image specified in the BOOT variable. As long as the switch can successfully boot from this image, the same image will be used on a reboot. If the switch fails to boot from the image specified in the BOOT variable, the switch will try to boot from the next image listed in the BOOT variable. If the end of the BOOT variable is reached without the switch booting successfully, the switch attempts the boot from the beginning of the BOOT variable. The autoboot continues until the switch successfully boots from one of the images specified in the BOOT variable.

Modifying the Boot Field and Using the boot Command

The configuration register boot field determines whether the switch loads an operating system image and, if so, where it obtains this system image. The following sections describe how to use and set the configuration register boot field and the procedures you must perform to modify the configuration register boot field. In ROMMON, you can use the **confreg** command to modify the configuration register and change boot settings.

Bits 0 through 3 of the software configuration register contain the boot field.



Note

The factory default configuration register setting for systems and spares is 0x2101. However, the recommended value is 0x0102.

When the boot field is set to either 00 or 01 (0-0-0-0 or 0-0-0-1), the system ignores any boot instructions in the system configuration file and the following occurs:

- When the boot field is set to 00, you must boot up the operating system manually by issuing the **boot** command at the system bootstrap or ROMMON prompt.
- When the boot field is set to 01, the system boots the first image in the bootflash single in-line memory module (SIMM).
- When the entire boot field equals a value between 0-0-1-0 and 1-1-1-1, the switch loads the system image specified by **boot system** commands in the startup configuration file.

**Caution**

If you set bootfield to a value between 0-0-1-0 and 1-1-1-1, you must specify a value in the **boot system** command, else the switch cannot boot up and will remain in ROMMON.

You can enter the **boot** command only or enter the command and include additional boot instructions, such as the name of a file stored in flash memory, or a file that you specify for booting from a network server. If you use the **boot** command without specifying a file or any other boot instructions, the system boots from the default flash image (the first image in onboard flash memory). Otherwise, you can instruct the system to boot up from a specific flash image (using the **boot system flash filename** command).

You can also use the **boot** command to boot up images stored in the compact flash cards located in slot 0 on the supervisor engine.

Modifying the Boot Field

Modify the boot field from the software configuration register.

To modify the software configuration register boot field, perform this task:

	Command	Purpose
Step 1	Switch# show version	Determines the current configuration register setting.
Step 2	Switch# configure terminal	Enters configuration mode, and specify the terminal option.
Step 3	Switch(config)# config-register value	Modifies the existing configuration register setting to reflect the way you want the switch to load a system image.
Step 4	Switch(config)# end	Exits configuration mode.
Step 5	Switch# reload	Reboots the switch to make your changes take effect.

To modify the configuration register while the switch is running Cisco IOS software, follow these steps:

Step 1 Enter the **enable** command and your password to enter privileged level, as follows:

```
Switch> enable
Password:
Switch#
```

Step 2 Enter the **configure terminal** command at the EXEC mode prompt (#), as follows:

```
Switch# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Switch(config)#
```

Step 3 Configure the configuration register to 0x102 as follows:

```
Switch(config)# config-register 0x102
```

Set the contents of the configuration register by specifying the *value* command variable, where *value* is a hexadecimal number preceded by 0x (see [Table 3-3 on page 3-27](#)).

Step 4 Enter the **end** command to exit configuration mode. The new value settings are saved to memory; however, the new settings do not take effect until the system is rebooted.

Step 5 Enter the **show version EXEC** command to display the configuration register value currently in effect; it will be used at the next reload. The value is displayed on the last line of the screen display, as shown in this sample output:

```
Configuration register is 0x141 (will be 0x102 at next reload)
```

Step 6 Save your settings.

See the “[Saving the Running Configuration Settings to Your Start-Up File](#)” section on page 3-10. Note that configuration register changes take effect only after the system reloads, such as when you enter a **reload** command from the console.

Step 7 Reboot the system. The new configuration register value takes effect with the next system boot up.

Verifying the Configuration Register Setting

Enter the **show version EXEC** command to verify the current configuration register setting. In ROMMON mode, enter the **show version** command to verify the configuration register setting.

To verify the configuration register setting for the switch, perform this task:

Command	Purpose
Switch# show version	Displays the configuration register setting.

In this example, the **show version** command indicates that the current configuration register is set so that the switch does not automatically load an operating system image. Instead, it enters ROMMON mode and waits for you to enter ROM monitor commands.

```
Switch# show version
Switch#sh version
Cisco IOS Software, IOS-XE Software, Catalyst 4500 L3 Switch Software
(cat4500e-UNIVERSAL-M), Version 03.02.00.SG2-2.193 CISCO INTERNAL USE ONLY UNIVERSAL
DEVELOPMENT K10 IOSD VERSION , synced to FLOGSBU8_122_56_SG_268
Copyright (c) 1986-2012 by Cisco Systems, Inc.
Compiled Tue 24-Jan-12 04:03 by sshivaga
Image text-base: 0x100DF3D4, data-base: 0x14B59A20

....

ROM: 15.0(1r)SG(0.326)
Jawa Revision 7, Snowtrooper Revision 0x0.0x116
```

```

Switch uptime is 44 minutes
System returned to ROM by power-on
System image file is
"tftp://172.25.60.31/auto/gsg-sw/interim/young/newest_image/iosxe/dev/cat4500e-universal.b
in"

License Information for 'WS-X45-SUP7-E'
  License Level: entservices   Type: Permanent
  Next reboot license Level: entservices

cisco WS-C4503-E (MPC8572) processor (revision 8) with 2097152K/20480K bytes of memory.
Processor board ID SPE134600QA
MPC8572 CPU at 1.5GHz, Supervisor 7
Last reset from PowerUp
1 Virtual Ethernet interface
96 Gigabit Ethernet interfaces
4 Ten Gigabit Ethernet interfaces
511K bytes of non-volatile configuration memory.

Configuration register is 0x40

```

¹Specifying the Startup System Image

You can enter multiple boot commands in the startup configuration file or in the BOOT environment variable to provide backup methods for loading a system image.

The BOOT environment variable is also described in the “Specify the Startup System Image in the Configuration File” section in the “Loading and Maintaining System Images and Microcode” chapter of the *Cisco IOS Configuration Fundamentals Configuration Guide*.

Use the following sections to configure your switch to boot from flash memory. flash memory can be either single in-line memory modules (SIMMs) or flash disks. Check the appropriate hardware installation and maintenance guide for information about types of flash memory.

Flash Memory Features

Flash memory allows you to do the following:

- Remotely load multiple system software images through TFTP or RCP transfers (one transfer for each file loaded)
- Boot a switch manually or automatically from a system software image stored in flash memory (you can also boot directly from ROM)
- Copy the system image to flash memory using TFTP
- Boot the system from flash memory either automatically or manually
- Copy the flash memory image to a network server using TFTP or RCP

For more information on Flash Memory, see this URL:

http://www.cisco.com/en/US/docs/switches/lan/catalyst4500/hardware/configuration/notes/OL_2788.html

Security Precautions

Note the following security precaution when loading from flash memory:

**Caution**

You can only change the system image stored in flash memory from privileged EXEC level on the console terminal.

Configuring Flash Memory

To configure your switch to boot from flash memory, perform the following procedure. (Refer to the appropriate hardware installation and maintenance publication for complete instructions on installing the hardware.)

-
- Step 1** Copy a system image to flash memory using TFTP or other protocols. Refer to the “Cisco IOS File Management” and “Loading and Maintaining System Images” chapters in the *Cisco IOS Configuration Fundamentals Configuration Guide*, Release 12.2, at the following URL:
- http://www.cisco.com/en/US/docs/ios/fundamentals/configuration/guide/12_2sr/cf_12_2sr_book.html
- Step 2** Configure the system to boot automatically from the desired file in flash memory.
- You might need to change the configuration register value. See the “[Modifying the Boot Field and Using the boot Command](#)” section on page 3-27, for more information on modifying the configuration register.
- Step 3** Save your configurations.
- Step 4** Power cycle and reboot your system to verify that all is working as expected.
-

Controlling Environment Variables

Although the ROM monitor controls environment variables, you can create, modify, or view them with certain commands. To create or modify the BOOT and BOOTLDR variables, use the **boot system and boot bootldr** global configuration commands, respectively. Refer to the “Specify the Startup System Image in the Configuration File” section in the “Loading and Maintaining System Images and Microcode” chapter of the *Configuration Fundamentals Configuration Guide* for details on setting the BOOT environment variable.

**Note**

When you use the **boot system and boot bootldr** global configuration commands, you affect only the running configuration. To save the configuration for future use, you must save the environment variable settings to your startup configuration, which places the information under ROM monitor control. Enter the **copy system:running-config nvram:startup-config** command to save the environment variables from your running configuration to your startup configuration.

You can view the contents of the BOOT and BOOTLDR variables using the **show bootvar** command. This command displays the settings for these variables as they exist in the startup configuration and in the running configuration if a running configuration setting differs from a startup configuration setting. This example shows how to check the BOOT and BOOTLDR variables on the switch:

```
Switch# show bootvar
BOOTLDR variable = bootflash:cat4000-is-mz,1;
```

```
Configuration register is 0x0
Switch#
```

Resetting a Switch to Factory Default Settings

Manufacturing and repair centers can use the **erase /all non-default** command to do the following:

- Clear the non-volatile configurations and states of the local supervisor engine (NVRAM and flashes).
- Set the factory default parameters on the Catalyst 4500 series switch before it is ready to ship to a customer.

For example, entering this command can generate the following output:

```
Switch# erase /all non-default
Erase and format operation will destroy all data in non-volatile storage. Continue?
[confirm]
Formatting bootflash: ...

Format of bootflash complete
Erasing nvram:
Erasing cat4000_flash:
Clearing crashinfo:data
Clearing the last power failure timestamp
Clearing all ROMMON variables
Setting default ROMMON variables:
    ConfigReg=0x2101
    PS1=rommon ! >
    EnableAutoConfig=1
Setting vtp mode to transparent
%WARNING! Please reboot the system for the changes to take effect
Switch#
00:01:48: %SYS-7-NV_BLOCK_INIT: Initialized the geometry of nvram
Switch#
```

If the Catalyst 4500 series switch is accessible to a TFTP server, you can copy an image to the bootflash memory with the TFTP command:

```
Switch# copy tftp://192.20.3.123/tftpboot/abc/cat4500-entservices-mz.bin bootflash:
```

When the copying is completed, you can reboot the just-copied Catalyst 4500 series switch image to the image stored in the bootflash memory with the **reload** command:

```
Switch# reload

System configuration has been modified. Save? [yes/no]: no
Proceed with reload? [confirm]

00:06:17: %SYS-5-RELOAD: Reload requested by console. Reload Reason: Reload Command.
```

To see details about the default parameters set by the **erase /all non-default** command, see the usage guidelines for the **erase** command in the *Catalyst 4500 Series Switch Cisco IOS Command Reference*.

Configuring Crashinfo

The crashinfo file is a collection of useful debugging information related to a process that has crashed. A system might have numerous processes running, and occasionally, one or more of them may crash

because of a software defect. The crashinfo file not only contain a snapshot of the process's state information of the process state at crash time, but it also contains a snapshot of the system's state information (such as the state of system resources). The information in the crashinfo file helps to identify a crash's possible root cause.

This section contains the following subsections:

- [About Crashinfo, page 3-33](#)
- [Configuring Crashinfo, page 3-35](#)

About Crashinfo

This subsection describes the three types of crashinfo files (process crashinfo, process core dump, and kernel crashinfo).

The following topics are discussed:

- [Process Crashinfo File, page 3-33](#)
- [Process Core Dump File, page 3-34](#)
- [Kernel Crashinfo File, page 3-34](#)

Process Crashinfo File

The naming convention of the crashinfo filename is

crashinfo_<process_name>_YYYYMMDD-HHMMSS-UTC

By default, this process crashinfo file is always generated for a process crash.

Although the file's contents are in readable text format, some parts of the file can only be decoded by a specific Cisco debugging tool. For nearly all process crashes, this file is required for root cause analysis.

This file contains the following information:

- Software exception that cause the crash
- Identity summary of the process
- Register values at the time of crash
- Stack dump of each of the threads of the process
- Files that are opened and used at the time of crash
- Command line as the process was launched
- Environment variables and resource limits
- Memory addresses where executable code are loaded
- Status of memory usage and CPU usage for each of the threads
- Application specific debugging information
- System boot log message
- System up time and CPU usage
- List of kernel loadable modules in the system
- System devices and their status
- List of mounted filesystems

- System memory total, free, and usage information
- Information on memory zones and slabs, and their status
- Virtual memory usage
- Resources maintained by the kernel
- List of program identifiers, called TRACEKEYs

Process Core Dump File

The naming convention for the core dump file name is

```
fullcore_<process_name>_YYYYMMDD-HHMMSS-UTC
```

By default, this file is not generated for a process crash. It can be enabled with the **exception CLI** command.

The file is of compressed binary type and can only be decoded by a specific Cisco debugging tool. The uncompressed file of this type is basically a coredump, which is a copy of memory pages that are used by the process that crashed. You should provide the coredump file, if available, as part of debugging information for root cause analysis.

By default, compression is enabled. We do not recommend that you enable coredump to obtain an uncompressed coredump because it requires a considerable storage space and time to save to the storage device. Enabling the uncompressed crash data generation is better suited for those systems with faster and larger storage capacity.

Kernel Crashinfo File

The filename of the kernel crashinfo file is **koops.dat**.

By default, the kernel crashinfo dump is enabled and cannot be disabled. The kernel crashinfo, also known as Kernel OOPS (KOOOPS), represents a defect in the kernel software code (a detected undesirable hardware condition). The term OOPS is an interjection made after a mistake.

Although the file's contents are in readable text format, some parts of this file can only be decoded by a trained engineer. For nearly all kernel crashes, this file is required for root cause analysis.

This file contains the following information:

- Exception summary
- CPU information
- List of kernel loadable modules
- Register values at the time of kernel oops event taken place
- Task that was running at the time of oops event
- List of code blocks executed leading up to the oops event
- A number of code instructions
- Time the oops event taken place

Configuring Crashinfo

The crashinfo is configured with default values. It is not necessary to configure the crashinfo because the default values are designed to be optimal. The configuration may require customization to account for system configuration, storage availability, and capacity.

Topics include:

- [Default Configuration, page 3-35](#)
- [Configuring commands, page 3-35](#)
- [show Commands, page 3-36](#)
- [Enabling the Generation of the Process Core Dump File, page 3-38](#)
- [Saving Crash Files to a Secondary Devices, page 3-39](#)
- [Determining the Process that Crashed, page 3-39](#)

Default Configuration

By default, the crash files are saved in crashinfo: partition, which cannot be changed. Two additional storage devices can be configured to save the crash files.

Using the **crashinfo** configuration commands, you can set the maximum number of files stored in each of the storage devices. This allows you to use the storage device wisely, saving the crash files, and increasing their manageability. By default, you can save up to 10 crashinfo files and up to 10 coredump files. Recall that the generation of the process core dump file is disabled by default. When the maximum number of files has been achieved and a new crash file is created, one or more of the oldest files, among the process crashinfo and core dump files, are deleted to create space for the new crash file.

The file named **last_crashinfo** possesses the file name of the latest crash file. View this file when it is hard to locate the latest crash file because multiple crash files exist on multiple storage devices.

Configuring commands

The following commands define the number of crashinfo files or coredump files respectively to be retained on each of the storage partition such as crashinfo:. Depending on the size of the storage partition, you might not be able to retain the configured number of files. Also, when a new crash file is created, and if there is insufficient space to store the new crash file, the oldest crash files are deleted until enough storage space is available for the new crash file.

Command	Purpose
Switch# exception crashinfo maximum-files ? <1-20> A value between 1-20	Sets the maximum number of process crashinfo files to be retained. The initial default values is 10.
Switch# exception coredump maximum-files ? <1-20> A value between 1-20	Sets the maximum number of process core dump files to be retained. The initial default values are 10.

The following command is used to configure the list of storage devices to be searched for space defined by the system when space is required to store newly created crash files. The first device is always the crashinfo: partition and cannot be changed. For example, if a second device is configured and there is insufficient space to store the new crash files, the system attempts to save these new crash files on the second device rather than deleting files from the crashinfo: partition.

Command	Purpose
Switch# exception dump device ? second Second search device to store crashinfo third Third search device to store crashinfo	Sets the second and third storage devices that are checked for available storage space for saving the new crash files. crashinfo: is the default device and cannot be changed,

The following command enables the generation of coredump crash data. The coredump data is a binary data, and it is compressed by default.

Command	Purpose
Switch# exception coredump	Enables the generation of compressed process core dump file. The default is not to generate compressed process core dump file.

The following command directs the system to generate uncompressed coredump crash data. Typically, you should not generate uncompressed coredump data because the file can very large (on the order of 1-2 Gigabytes), and storage sapce for the crash file will be insufficient.

Command	Purpose
Switch# exception coredump uncompress	Enables the generation of the uncompressed process core dump file. Note An uncompressed coredump file can be very large. The default is to generate compressed process core dump file.

show Commands

Use the following command to display the current configuration of the crashinfo:

```
Switch# show exception information
Exception configuration information
Coredump file - disabled,compressed
Maximum number of files
    Core - 10 file(s)
    Process crashinfo - 10 file(s)
Configured storage devices
    1 - crashinfo:
    2 - not assigned
    3 - not assigned
```

Dump protocol - not configured

Use the following command to determine the file name of latest crash file:

```
Switch# more crashinfo:last_crashinfo
crashinfo:crashinfo_iosd_20100803-073408-UTC
```

Use the following command to view the contents of a process crashinfo file:

```
Switch# more crashinfo:crashinfo_iosd_20100803-073408-UTC
===== signr 5
```

===== Process

```
PID=11773   PPPID=1   PGRP=11773
Time: User=7794   System=724
```

===== current - pid 11773

===== Registers

```
nip = 0x0C3CC798   msr = 0x0002D000   r3* = 0x00002DFD   ctr = 0x0CF05040
lnk = 0x0CF05050   xer = 0x20000000   ccr = 0x42002444
trp = 0x00000C00   dar = 0x143CDC68   isr = 0x00000000   res = 0x00000000
R0  0x000000FA   R1  0x7B37D090   R2  0x4804D780   R3  0x00000000
R4  0x00002DFD   R5  0x00000005   R6  0x143CDC68   R7  0x00000000
R8  0x00000000   R9  0x00002DFD   R10 0x7B37CEA8   R11 0x0CF05040
R12 0x22002448   R13 0x14FF4008   R14 0x10B50F44   R15 0x00000000
R16 0x00000000   R17 0x14CB665C   R18 0x13E70000   R19 0x14CB0000
R20 0x14FF0000   R21 0x00000001   R22 0x891753A1   R23 0x891752A0
R24 0x15AB30D8   R25 0x00000000   R26 0x00000000   R27 0x143CDC68
R28 0x000007CA   R29 0x7B37D0E0   R30 0x143D10C4   R31 0x7820B3D4
```

--More--

//output omitted//

Use the following command to view the contents of kernel crash information

```
Switch# more crashinfo:koops.dat
NOVAOOPS:
<4>CRASHINFO_OOPS:
<4>Oops: Machine check, sig: 7 [#1]
<4>SMP NR_CPUS=2 Cisco Systems K10 (Callista)
<4>Modules linked in: gsbu64atomic mtsmod(P) aipcmmod(P) vsi uld_timer cpumem tun bridge
pds crashinfo bpa_mem
<4>NIP: c0022cc0 LR: c00230e8 CTR: c00230b0
<4>REGS: c04c4f50 TRAP: 0202 Tainted: P (2.6.24.4.28.13.k10)
<4>MSR: 00021000 <ME> CR: 24084422 XER: 20000000
<4>TASK = ea088150[13355] 'iosd' THREAD: ea738000 CPU: 0
<6>GPR00: 00000001 ea739c60 ea088150 0000001a ea4c0460 00000000 00000000 00000000
<6>GPR08: c0fec000 c04d0000 fcfd3000 fcfd3060 24084482 1423ebb8 00000000 10121d28
<6>GPR16: 100d0000 ea739db8 c03f61a0 c03fb098 ea088318 c04c0000 ea739f50 ea088530
<6>GPR24: 418004fc ea739e88 00029000 00021000 c04c8eb0 0000005a 0000005a e9b80fa0
<4>NIP [c0022cc0] cisco_tatooine_mask_irq_internal+0x10/0x40
<4>LR [c00230e8] cisco_tatooine_mask_and_ack_irq+0x38/0x60
<4>Call Trace:
<4>[ea739c60] [c00f3004] proc_destroy_inode+0x24/0x40 (unreliable)
<4>[ea739c80] [c0073e0c] free_irq+0x14c/0x170
<4>[ea739ca0] [c0075ee0] irq_proc_release+0x50/0x170
<4>[ea739cc0] [c00f3570] proc_reg_release+0x70/0xd0
<4>[ea739ce0] [c00ab28c] __fput+0xbc/0x1b0
<4>[ea739d00] [c00a777c] filp_close+0x6c/0xd0
<4>[ea739d20] [c004506c] put_files_struct+0x10c/0x120
<4>[ea739d40] [c00468f8] do_exit+0x198/0x7a0
<4>[ea739d90] [c0046f38] do_group_exit+0x38/0x90
```

```

<4>[ea739db0] [c0053cc8] get_signal_to_deliver+0x3a8/0x610
<4>[ea739e80] [c0010d68] do_signal+0x48/0x2f0
<4>[ea739f40] [c0018a64] do_user_signal+0x74/0xc4
<4>Instruction dump:
<4>38000001 7c00e830 7d400378 5400043e 7c0004ac b00b0100 4bffffac 00000000
<4>3d20c04d 81498eac 396a0060 7c0004ac <812a0060> 0c090000 4c00012c 3800ffffe

```

```

TIMESTAMP:
730956298.010309
READFLAG:
1

```

Use the following command to list the current crash files on the system:

```

Switch# show exception files
Exception crashinfo files

```

```

Most Recent Crashinfo file:
crashinfo:crashinfo_iosd_20100803-073408-UTC

```

```

Files in crashinfo:
crashinfo_iosd_20100803-064137-UTC
crashinfo_iosd_20100803-073408-UTC

```

Enabling the Generation of the Process Core Dump File

If there might be insufficient storage space, you might want to reduce the number of process core dump files. To do that you must first enable the generation of the process core dump file by performing the following task:

	Command	Purpose
Step 1	Switch# config terminal	Enters global configuration mode.
Step 2	Switch(config)# exception coredump	Generates the process core dump file.
Step 3	Switch(config)# exception coredump maximum-files x	Limits the number of process core dump files to <i>x</i> .
Step 4	Switch(config)# end	Returns to global configuration mode.
Step 5	Switch# show exception information	(optional) Verifies the configuration.

This example shows how to generate the process core dump file, limit the number of process core dump files to 2, and to verify the configuration:

```

Switch# config t
Switch (config)# exception coredump
Switch (config)# exception coredump maximum files 2
Switch (config)# end
Switch# show exception information
Exception configuration information
Core dump file - enabled, compressed
Maximum number of files
    Core - 2 file(s)
    Process crashinfo - 10 file(s)
Configured storage devices
    1 - crashinfo:
    2 - not assigned
    3 - not assigned
Dump protocol - not configured

```

Saving Crash Files to a Secondary Devices

Assume that the system has an installed USB storage device and you want to save the crash files to the USB storage instead of deleting the old crash files from the default storage crashinfo: partition. The system starts saving the crash files to either installed and configured USB, when either the maximum number of files had been reached or there is insufficient space on the crashinfo: partition.

To configure to save a crash file in a secondary device, perform the following task:

	Command	Purpose
Step 1	Switch# config terminal	Enters global configuration mode.
Step 2	Switch(config)# exception dump second device x	Configures <i>x</i> as the storage devices to be searched for needed storage space.
Step 3	Switch(config)# end	Returns to global configuration mode.
Step 4	Switch# show exception information	(optional) Verifies the configuration. Note A portion of the output should indicate that the secondary device has been configured. Configured storage devices 1 - crashinfo: 2 - usb0: 3 - not assigned

This example shows how to configure the USB to be searched for needed storage space and to verify the configuration>

```
Switch# config t
Switch(config)# exception dump second device usb0:
Switch# show exception information
Exception configuration information
Coredump file - disabled,compressed
Maximum number of files
    Core - 10 file(s)
    Process crashinfo - 10 file(s)
Configured storage devices
    1 - crashinfo:
    2 - usb0:
    3 - not assigned
Dump protocol - not configured
```

Determining the Process that Crashed

Your system might have rebooted after a critical process had crashed. If so, you might want to determine the location of the latest crash file, the process that crashed, and the likely root cause. Cisco TAC requires the available crash files for root cause analysis. If the crash is due to some side-effects of another process that crashed earlier, the latest crash file may not contain the actual root cause information. We recommend that you provide Cisco TAC all the crash files that are generated around crash time.

To determine the process that crashed, perform this task:

	Command	Purpose
Step 1	Switch# show exception information	Displays the configuration to determine where the crash files are saved.
Step 2	Switch# more crashinfo:last_crashinfo	Locates the latest crashinfo file.
Step 3	Switch# more crashinfo:crashinfo_yyyyymmdd-hhmmss-UTC	Displays the contents of the process crashinfo file.
Step 4	Switch(config)# more crashinfo:koops.dat	Displays the contents of the kernel crashinfo file to determine whether the kernel had crashed.
Step 5	Switch# copy crashinfo:tftp: Source filename []? Address or name of remote host []? remote_host Destination filename []? crashinfo_yyyyymmdd-hhmmss-UTC	Copies a crash file to a TFTP server. When Cisco TAC requests a specific crash file for root cause analysis, you can copy the file to a TFTP server from the switch.

The following examples show how to use these commands:

```
Switch# show exception information
Exception configuration information
Coredump file - disabled,compressed
Maximum number of files
    Core - 10 file(s)
    Process crashinfo - 10 file(s)
Configured storage devices
    1 - crashinfo:
    2 - not assigned
    3 - not assigned
Dump protocol - not configured

Switch# more crashinfo:last_crashinfo
crashinfo:crashinfo_iosd_20100803-073408-UTC

Switch# more crashinfo:crashinfo_iosd_20100803-073408-UTC
===== signr 5

===== Process

PID=11773  PPPID=1  PGRP=11773
Time: User=7794  System=724

===== current - pid 11773

===== Registers

nip = 0x0C3CC798  msr = 0x0002D000  r3* = 0x00002DFD  ctr = 0x0CF05040
lnk = 0x0CF05050  xer = 0x20000000  ccr = 0x42002444
trp = 0x00000C00  dar = 0x143CDC68  isr = 0x00000000  res = 0x00000000
R0 0x000000FA  R1 0x7B37D090  R2 0x4804D780  R3 0x00000000
R4 0x00002DFD  R5 0x00000005  R6 0x143CDC68  R7 0x00000000
R8 0x00000000  R9 0x00002DFD  R10 0x7B37CEA8  R11 0x0CF05040
R12 0x22002448  R13 0x14FF4008  R14 0x10B50F44  R15 0x00000000
R16 0x00000000  R17 0x14CB665C  R18 0x13E70000  R19 0x14CB0000
R20 0x14FF0000  R21 0x00000001  R22 0x891753A1  R23 0x891752A0
R24 0x15AB30D8  R25 0x00000000  R26 0x00000000  R27 0x143CDC68
R28 0x000007CA  R29 0x7B37D0E0  R30 0x143D10C4  R31 0x7820B3D4

--More-

//output omitted//
```



```

Switch# more crashinfo:koops.dat
NOVAOOPS:
<4>CRASHINFO_OOPS:
<4>Oops: Machine check, sig: 7 [#1]
<4>SMP NR_CPUS=2 Cisco Systems K10 (Callista)
<4>Modules linked in: gsibu64atomic mtsmod(P) aipcmmod(P) vsi uld_timer cpumem tun bridge
pds crashinfo bpa_mem
<4>NIP: c0022cc0 LR: c00230e8 CTR: c00230b0
<4>REGS: c04c4f50 TRAP: 0202 Tainted: P (2.6.24.4.28.13.k10)
<4>MSR: 00021000 <ME> CR: 24084422 XER: 20000000
<4>TASK = ea088150[13355] 'iosd' THREAD: ea738000 CPU: 0
<6>GPR00: 00000001 ea739c60 ea088150 0000001a ea4c0460 00000000 00000000 00000000
<6>GPR08: c0fec000 c04d0000 fcfd3000 fcfd3060 24084482 1423ebb8 00000000 10121d28
<6>GPR16: 100d0000 ea739db8 c03f61a0 c03fb098 ea088318 c04c0000 ea739f50 ea088530
<6>GPR24: 418004fc ea739e88 00029000 00021000 c04c8eb0 0000005a 0000005a e9b80fa0
<4>NIP [c0022cc0] cisco_tatooine_mask_irq_internal+0x10/0x40
<4>LR [c00230e8] cisco_tatooine_mask_and_ack_irq+0x38/0x60
<4>Call Trace:
<4>[ea739c60] [c00f3004] proc_destroy_inode+0x24/0x40 (unreliable)
<4>[ea739c80] [c0073e0c] free_irq+0x14c/0x170
<4>[ea739ca0] [c0075ee0] irq_proc_release+0x50/0x170
<4>[ea739cc0] [c00f3570] proc_reg_release+0x70/0xd0
<4>[ea739ce0] [c00ab28c] __fput+0xbc/0x1b0
<4>[ea739d00] [c00a777c] filp_close+0x6c/0xd0
<4>[ea739d20] [c004506c] put_files_struct+0x10c/0x120
<4>[ea739d40] [c00468f8] do_exit+0x198/0x7a0
<4>[ea739d90] [c0046f38] do_group_exit+0x38/0x90
<4>[ea739db0] [c0053cc8] get_signal_to_deliver+0x3a8/0x610
<4>[ea739e80] [c0010d68] do_signal+0x48/0x2f0
<4>[ea739f40] [c0018a64] do_user_signal+0x74/0xc4
<4>Instruction dump:
<4>38000001 7c00e830 7d400378 5400043e 7c0004ac b00b0100 4bffffac 00000000
<4>3d20c04d 81498eac 396a0060 7c0004ac <812a0060> 0c090000 4c00012c 3800ffff

TIMESTAMP:
730956298.010309
READFLAG:
Switch# copy crashinfo: tftp:
Source filename []?
Address or name of remote host []? 10.5.224.2
Destination filename []? /users/user/crashinfo_iosd_20100803-073408-UTC
!!!!!!
790676 bytes copied in 37.420 secs (21130 bytes/sec)

```

