

Switch Memory and Configuration

Contents

Overview	6-3
Configuration File Management	6-3
Using the CLI To Implement Configuration Changes	6-6
Using the Menu and Web Browser Interfaces To Implement Configuration Changes	6-10
Menu: Implementing Configuration Changes	6-10
Using Save and Cancel in the Menu Interface	6-10
Rebooting from the Menu Interface	6-11
Web: Implementing Configuration Changes	6-13
Using Primary and Secondary Flash Image Options	6-14
Displaying the Current Flash Image Data	6-14
Switch Software Downloads	6-16
Local Switch Software Replacement and Removal	6-17
Rebooting the Switch	6-19
Operating Notes about Booting	6-19
Boot and Reload Command Comparison	6-20
Setting the Default Flash	6-21
Booting from the Default Flash (Primary or Secondary)	6-22
Booting from a Specified Flash	6-23
Using Reload	6-24
Multiple Configuration Files	6-26
General Operation	6-27
Transitioning to Multiple Configuration Files	6-29
Listing and Displaying Startup-Config Files	6-30
Viewing the Startup-Config File Status with Multiple Configuration Enabled	6-30
Displaying the Content of A Specific Startup-Config File	6-31

Changing or Overriding the Reboot Configuration Policy	6-31
Managing Startup-Config Files in the Switch	6-33
Renaming an Existing Startup-Config File	6-34
Creating a New Startup-Config File	6-34
Erasing a Startup-Config File	6-35
Using the Clear + Reset Button Combination To Reset the Switch to Its Default Configuration	6-37
Transferring Startup-Config Files To or From a Remote Server	6-38
TFTP: Copying a Configuration File to a Remote Host	6-38
TFTP: Copying a Configuration File from a Remote Host	6-39
Xmodem: Copying a Configuration File to a Serially Connected Host	6-39
Xmodem: Copying a Configuration from a Serially Connected Host	6-40
Operating Notes for Multiple Configuration Files	6-40

Overview

This chapter describes:

- How switch memory manages configuration changes
 - How the CLI implements configuration changes
 - How the menu interface and web browser interface implement configuration changes
 - How the switch provides software options through primary/secondary flash images
 - How to use the switch's primary and secondary flash options, including displaying flash information, booting or restarting the switch, and other topics
-

Configuration File Management

The switch maintains two configuration files, the *running-config* file and the *startup-config* file.

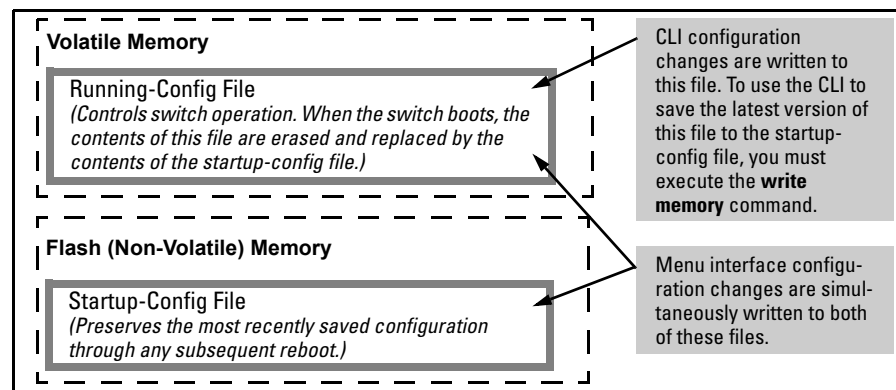


Figure 6-1. Conceptual Illustration of Switch Memory Operation

- **Running Config File:** Exists in volatile memory and controls switch operation. If no configuration changes have been made in the CLI since the switch was last booted, the running-config file is identical to the startup-config file.
-

- **Startup-config File:** Exists in flash (non-volatile) memory and is used to preserve the most recently-saved configuration as the “permanent” configuration.

Booting the switch replaces the current running-config file with a new running-config file that is an exact copy of the current startup-config file.

Note

Any of the following actions boots the switch:

- Executing the **boot** or the **reload** command in the CLI
- Executing the **boot** command in the menu interface
- Pressing the Reset button on the front of the switch
- Removing, then restoring power to the switch

For more on reboots and the switch’s dual-flash images, refer to “Using Primary and Secondary Flash Image Options” on page 6-14.

Options for Saving a New Configuration. Making one or more changes to the running-config file creates a new operating configuration. *Saving* a new configuration means to overwrite (replace) the current startup-config file with the current running-config file. This means that if the switch subsequently reboots for any reason, it will resume operation using the new configuration instead of the configuration previously defined in the startup-config file. There are three ways to save a new configuration:

- **In the CLI:** Use the **write memory** command. This overwrites the current startup-config file with the contents of the current running-config file.
- **In the menu interface:** Use the **Save** command. This overwrites *both* the running-config file and the startup-config file with the changes you have specified in the menu interface screen.
- **In the web browser interface:** Use the **[Apply Changes]** button or other appropriate button. This overwrites *both* the running-config file and the startup-config file with the changes you have specified in the web browser interface window.

Note that using the CLI instead of the menu or web browser interface gives you the option of changing the running configuration without affecting the startup configuration. This allows you to test the change without making it “permanent”. When you are satisfied that the change is satisfactory, you can make it permanent by executing the **write memory** command. For example, suppose you use the following command to disable port 5:

```
ProCurve(config)# interface ethernet 5 disable
```

The above command disables port 5 in the running-config file, but not in the startup-config file. Port 5 remains disabled only until the switch reboots. If you want port 5 to remain disabled through the next reboot, use **write memory** to save the current running-config file to the startup-config file in flash memory.

```
ProCurve(config)# write memory
```

If you use the CLI to make a configuration change and then change from the CLI to the Menu interface without first using write memory to save the change to the startup-config file, then the switch prompts you to save the change. For example, if you use the CLI to create VLAN 20, and then select the menu interface, VLAN 20 is configured in the running-config file, but not in the startup-config file. In this case you will see:

```
ProCurve(config)# vlan 20
ProCurve(config)# menu
Do you want to save current configuration [y/n]?
```

If you type **[Y]**, the switch overwrites the startup-config file with the running-config file, and your configuration change(s) will be preserved across reboots. If you type **[N]**, your configuration change(s) will remain only in the running-config file. In this case, if you do not subsequently save the running-config file, your unsaved configuration changes will be lost if the switch reboots for any reason.

Storing and Retrieving Configuration Files. You can store or retrieve a backup copy of the startup-config file on another device. For more information, refer to the section on “Transferring Switch Configurations” on page A-24 in Appendix A on “File Transfers”.

USB Autorun. This feature supports the ability to auto execute CLI commands stored on a USB flash drive (for example, to configure the switch, update software, retrieve diagnostics, etc.). For more information, refer to the section on “Using USB Autorun” on page A-37.

Using the CLI To Implement Configuration Changes

The CLI offers these capabilities:

- Access to the full set of switch configuration features
- The option of testing configuration changes before making them permanent

How To Use the CLI To View the Current Configuration Files. Use **show** commands to view the configuration for individual features, such as port status or Spanning Tree Protocol. However, to view either the entire startup-config file or the entire running-config file, use the following commands:

- **show config** — Displays a listing of the current startup-config file.
- **show running-config** — Displays a listing of the current running-config file.
- **write terminal** — Displays a listing of the current running-config file.
- **show config status** — Compares the startup-config file to the running-config file and lists one of the following results:
 - If the two configurations are the same you will see:
 - Running configuration is the same as the startup configuration.
 - If the two configurations are different, you will see:
 - Running configuration has been changed and needs to be saved.

Note

Show config, **show running-config**, and **write terminal** commands display the configuration settings that differ from the switch's factory-default configuration.

How To Use the CLI To Reconfigure Switch Features. Use this procedure to permanently change the switch configuration (that is, to enter a change in the startup-config file).

1. Use the appropriate CLI commands to reconfigure the desired switch parameters. This updates the selected parameters in the running-config file.
2. Use the appropriate **show** commands to verify that you have correctly made the desired changes.

3. Observe the switch's performance with the new parameter settings to verify the effect of your changes.
4. When you are satisfied that you have the correct parameter settings, use the **write memory** command to copy the changes to the startup-config file.

Syntax: write memory

Saves the running configuration file to the startup-config. The saved configuration becomes the boot-up configuration of the switch on the next boot.

When using redundant management, saves the running configuration of the switch to flash on the active management module. The saved configuration becomes the boot-up configuration of the switch the next time it is booted. The saved configuration file is sync'd to the standby management module.

Note: If the active management module and the standby management module are running on different operating systems because the boot set-default command was executed and then the standby module was rebooted, the write memory command displays this warning: "Warning: The next reboot or failover is set to boot from a different software image. These config changes may be incompatible or not used after a reboot or failover."

For example, the default port mode setting is **auto**. Suppose that your network uses Cat 3 wiring and you want to connect the switch to another autosensing device capable of 100 Mbps operation. Because 100 Mbps over Cat 3 wiring can introduce transmission problems, the recommended port mode is **auto-10**, which allows the port to negotiate full- or half-duplex, but restricts speed to 10 Mbps. The following command configures port A5 to auto-10 mode in the running-config file, allowing you to observe performance on the link without making the mode change permanent.

```
ProCurve(config)# interface e a5 speed-duplex auto-10
```

After you are satisfied that the link is operating properly, you can save the change to the switch's permanent configuration (the startup-config file) by executing the following command:

```
ProCurve(config)# write memory
```

The new mode (**auto-10**) on port A5 is now saved in the startup-config file, and the startup-config and running-config files are identical. If you subsequently reboot the switch, the **auto-10** mode configuration on port A5 will remain because it is included in the startup-config file.

How To Cancel Changes You Have Made to the Running-Config File.

If you use the CLI to change parameter settings in the running-config file, and then decide that you don't want those changes to remain, you can use either of the following methods to remove them:

- Manually enter the earlier values you had for the changed settings. (This is recommended if you want to restore a small number of parameter settings to their previous boot-up values.)
- Update the running-config file to match the startup-config file by rebooting the switch. (This is recommended if you want to restore a larger number of parameter settings to their previous boot-up values.)

If you use the CLI to change a parameter setting, and then execute the **boot** command without first executing the **write memory** command to save the change, the switch prompts you to specify whether to save the changes in the current running-config file. For example:

```
ProCurve(config)# interface e 1 disable
ProCurve(config)# boot
Device will be rebooted, do you want to continue [y/n]? y
Do you want to save current configuration [y/n]?
```

Figure 6-2. Boot Prompt for an Unsaved Configuration

The above prompt means that one or more parameter settings in the running-config file differ from their counterparts in the startup-config file and you need to choose which config file to retain and which to discard.

- If you want to update the startup-config file to match the running-config file, press **[Y]** for “yes”. (This means that the changes you entered in the running-config file will be saved in the startup-config file.)
- If you want to discard the changes you made to the running-config file so that it will match the startup-config file, then press **[N]** for “no”. (This means that the switch will discard the changes you entered in the running-config file and will update the running-config file to match the startup-config file.)

Note

If you use the CLI to make a change to the running-config file, you should either use the **write memory** command or select the save option allowed during a reboot (figure 6-6-2, above) to save the change to the startup-config file. That is, if you use the CLI to change a parameter setting, but then reboot the switch from either the CLI or the menu interface without first executing the **write memory** command in the CLI, the current startup-config file will replace the running-config file, and any changes in the running-config file will be lost.

Using the **Save** command in the menu interface does not save a change made to the running config by the CLI unless you have also made a configuration change in the menu interface. Also, the menu interface displays the current running-config values. Thus, where a parameter setting is accessible from both the CLI and the menu interface, if you change the setting in the CLI, the new value will appear in the menu interface display for that parameter. *However, as indicated above, unless you also make a configuration change in the menu interface, only the write memory command in the CLI will actually save the change to the startup-config file.*

How To Reset the startup-config and running-config Files to the Factory Default Configuration. This command reboots the switch, replacing the contents of the current startup-config and running-config files with the factory-default startup configuration.

Syntax: erase startup-config

For example:

```
ProCurve(config)# erase startup-config
Configuration will be deleted and device rebooted, continue [y/n]?
```

Figure 6-3. Example of erase startup-config Command

Press [y] to replace the current configuration with the factory default configuration and reboot the switch. Press [n] to retain the current configuration and prevent a reboot.

In a redundant management system, this command erases the startup config file on both the active and the standby management modules as long as redundancy has not been disabled. If the standby management module is not in standby mode or has failed selftest, the startup config file is not erased.

Using the Menu and Web Browser Interfaces To Implement Configuration Changes

The menu and web browser interfaces offer these advantages:

- Quick, easy menu or window access to a subset of switch configuration features
- Viewing several related configuration parameters in the same screen, with their default and current settings
- Immediately changing both the running-config file and the startup-config file with a single command

Menu: Implementing Configuration Changes

You can use the menu interface to simultaneously save and implement a subset of switch configuration changes without having to reboot the switch. That is, when you save a configuration change in the menu interface, you simultaneously change both the running-config file and the startup-config file.

Note

The only exception to this operation are two VLAN-related parameter changes that require a reboot—described under “Rebooting To Activate Configuration Changes” on page 6-12.

Using **Save** and **Cancel** in the Menu Interface

For any configuration screen in the menu interface, the Save command:

1. Implements the changes in the running-config file
2. Saves your changes to the startup-config file

If you decide not to save and implement the changes in the screen, select **Cancel** to discard them and continue switch operation with the current operation. For example, suppose you have made the changes shown below in the System Information screen:

To save and implement the changes for all parameters in this screen, press the **[Enter]** key, then press **[S]** (for **Save**). To cancel all changes, press the **[Enter]** key, then press **[C]** (for **Cancel**)

```

ProCurve
----- CONSOLE - MANAGER MODE -----
Switch Configuration - System Information

System Name : ProCurve Switch
System Contact :
System Location :

Inactivity Timeout (min) [0] : 0      MAC Age Time (sec) [300] : 300
Inbound Telnet Enabled [Yes] : Yes    Web Agent Enabled [Yes] : Yes
Time Sync Method [None] : TIMEP
TimeP Mode [Disabled] : Disabled

Time Zone [0] : 0
Daylight Time Rule [None] : Continental-US-and-Canada

Actions->  C Cancel      E Edit      S Save      H Help

Select Daylight Time Rule for your location.
Use arrow keys to change field selection, <Space> to toggle field choices,
and <Enter> to go to Actions.

```

Figure 6-4. Example of Pending Configuration Changes You Can Save or Cancel

Note

If you reconfigure a parameter in the CLI and then go to the menu interface without executing a **write memory** command, those changes are stored only in the running configuration (even if you execute a Save operation in the menu interface). If you then execute a switch **boot** command in the menu interface, the switch discards the configuration changes made while using the CLI. To ensure that changes made while using the CLI are saved, execute **write memory** in the CLI before rebooting the switch.

Rebooting from the Menu Interface

- Terminates the current session and performs a reset of the operating system
- Activates any configuration changes that require a reboot
- Resets statistical counters to zero

(Note that statistical counters can be reset to zero without rebooting the switch. See “To Display the Port Counter Summary Report” on page 16.)

To Reboot the switch, use the **Reboot Switch** option in the Main Menu. (Note that the Reboot Switch option is not available if you log on in Operator mode; that is, if you enter an Operator password instead of a manager password at the password prompt.)

Switch Memory and Configuration

Using the Menu and Web Browser Interfaces To Implement Configuration Changes

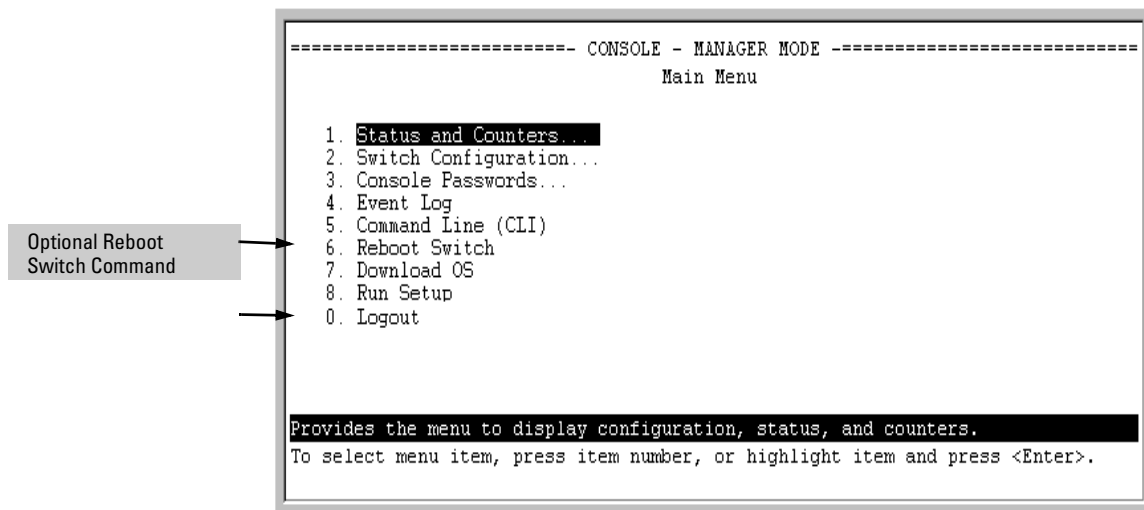


Figure 6-5. The Reboot Switch Option in the Main Menu

Rebooting To Activate Configuration Changes. Configuration changes for most parameters become effective as soon as you save them. However, you must reboot the switch in order to implement a change in the **Maximum VLANs to support** parameter.

(To access these parameters, go to the Main menu and select **2. Switch Configuration**, then **8. VLAN Menu**, then **1. VLAN Support**.)

If configuration changes requiring a reboot have been made, the switch displays an asterisk (*) next to the menu item in which the change has been made. For example, if you change and save parameter values for the **Maximum VLANs to support** parameter, an asterisk appears next to the **VLAN Support** entry in the VLAN Menu screen, and also next to the **Switch Configuration** ..entry in the Main menu, as shown in Figure 6-6:

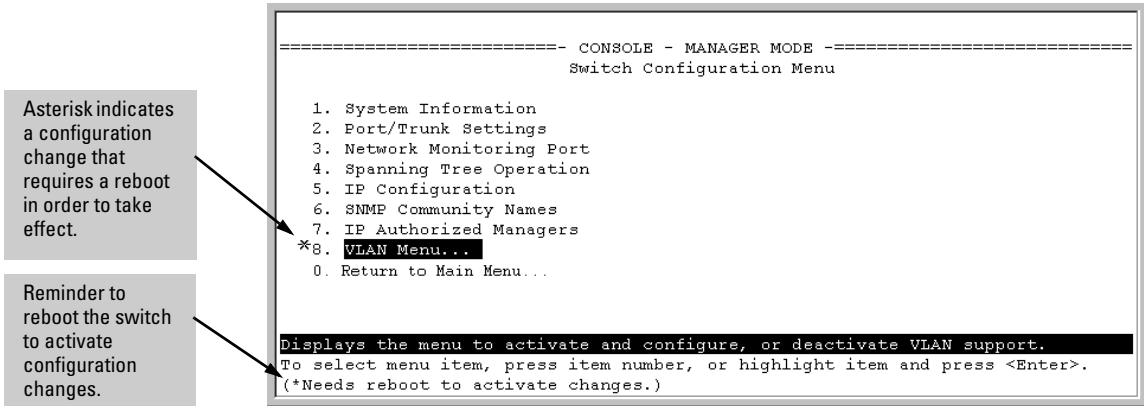


Figure 6-6. Indication of a Configuration Change Requiring a Reboot

Web: Implementing Configuration Changes

You can use the web browser interface to simultaneously save and implement a subset of switch configuration changes without having to reboot the switch. That is, when you save a configuration change (in most cases, by clicking on **[Apply Changes]** or **[Apply Settings]**), you simultaneously change both the running-config file and the startup-config file.

Note

If you reconfigure a parameter in the CLI and then go to the browser interface without executing a **write memory** command, those changes will be saved to the startup-config file if you click on **[Apply Changes]** or **[Apply Settings]** in the web browser interface.

Using Primary and Secondary Flash Image Options

The switches covered in this guide feature two flash memory locations for storing switch software image files:

- **Primary Flash:** The default storage for a switch software image.
- **Secondary Flash:** The additional storage for either a redundant or an alternate switch software image.

With the Primary/Secondary flash option you can test a new image in your system without having to replace a previously existing image. You can also use the image options for troubleshooting. For example, you can copy a problem image into Secondary flash for later analysis and place another, proven image in Primary flash to run your system. The switch can use only one image at a time.

The following tasks involve primary/secondary flash options:

- Displaying the current flash image data and determining which switch software versions are available
- Switch software downloads
- Replacing and removing (erasing) a local switch software version
- System booting

Displaying the Current Flash Image Data

Use the commands in this section to:

- Determine whether there are flash images in both primary and secondary flash
- Determine whether the images in primary and secondary flash are the same
- Identify which switch software version is currently running

Viewing the Currently Active Flash Image Version. This command identifies the software version on which the switch is currently running, and whether the active version was booted from the primary or secondary flash image.

Syntax:show version

For example, if the switch is using a software version of K.12.XX stored in Primary flash, **show version** produces the following:

```
ProCurve(config)# show version

Image stamp:   /su/code/build/info(s01)
               Dec 01 2006 10:50:26
               K.12.XX
               1223
Boot Image:    Primary
```

Figure 6-7. Example Showing the Identity of the Current Flash Image

Determining Whether the Flash Images Are Different Versions. If the flash image sizes in primary and secondary are the same, then in almost every case, the primary and secondary images are identical. This command provides a comparison of flash image sizes, plus the boot ROM version and from which flash image the switch booted. For example, in the following case, the images are different versions of the switch software, and the switch is running on the version stored in the secondary flash image:

```
ProCurve(config)# show flash
Image          Size(Bytes)   Date   Version  Build #
-----
Primary Image  : 7493854   03/21/07 K.12.29  1617
Secondary Image : 7463821   03/23/07 K.12.30  1700

Boot Rom Version: K.12.30
Default Boot    : Primary
```

Will boot from primary flash
on the next boot.

Figure 6-8. Example Showing Different Flash Image Versions

Determining Which Flash Image Versions Are Installed. The **show version** command displays which software version the switch is currently running and whether that version booted from primary or secondary flash. Thus, if the switch booted from primary flash, you will see the version number of the software version stored in primary flash, and if the switch booted from secondary flash, you will see the version number of the software version stored in secondary flash. Thus, by using **show version**, then rebooting the switch from the opposite flash image and using **show version** again, you can determine the version(s) of switch software in both flash sources. For example:

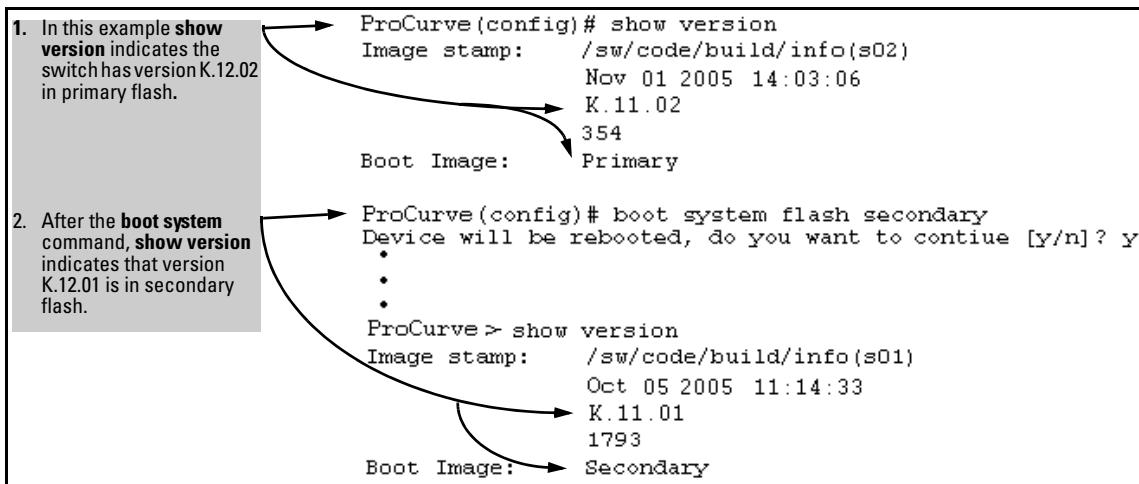


Figure 6-9. Determining the Software Version in Primary and Secondary Flash

Switch Software Downloads

The following table shows the switch’s options for downloading a software version to flash and booting the switch from flash

Table 6-1. Primary/Secondary Memory Access

Action	Menu	CLI	Web Browser	SNMP
Download to Primary	Yes	Yes	Yes	Yes
Download to Secondary	No	Yes	No	Yes
Boot from Primary	Yes	Yes	Yes	Yes
Boot from Secondary	No	Yes	No	Yes

The different software download options involve different **copy** commands, plus **xmodem**, **usb**, and **ftpp**. These topics are covered in Appendix A, “File Transfers”.

Download Interruptions. In most cases, if a power failure or other cause interrupts a flash image download, the switch reboots with the image previously stored in primary flash. In the unlikely event that the primary image is corrupted, as a result of an interruption, the switch will reboot from secondary flash and you can either copy the secondary image into primary or download another image to primary from an external source. Refer to Appendix A, “File Transfers”.

Local Switch Software Replacement and Removal

This section describes commands for erasing a software version and copying an existing software version between primary and secondary flash.

Note

It is not necessary to erase the content of a flash location before downloading another software file. The process automatically overwrites the previous file with the new file. If you want to remove an unwanted software version from flash, ProCurve recommends that you do so by overwriting it with the same software version that you are using to operate the switch, or with another acceptable software version. To copy a software file between the primary and secondary flash locations, refer to “Copying a Switch Software Image from One Flash Location to Another”, below.

The local commands described here are for flash image management within the switch. To download a software image file from an external source, refer to Appendix A, “File Transfers”.

Copying a Switch Software Image from One Flash Location to

Another. When you copy the flash image from primary to secondary or the reverse, the switch overwrites the file in the destination location with a copy of the file from the source location. This means you *do not* have to erase the current image at the destination location before copying in a new image.

Caution

Verify that there is an acceptable software version in the source flash location from which you are going to copy. Use the **show flash** command or, if necessary, the procedure under “Determining Which Flash Image Versions Are Installed” on page 6-15 to verify an acceptable software version. Attempting to copy from a source image location that has a corrupted flash image overwrites the image in the destination flash location. In this case, the switch will not have a valid flash image in either flash location, but will continue running on a temporary flash image in RAM. *Do not reboot the switch.* Instead, immediately download another valid flash image to primary or secondary flash. Otherwise, if the switch is rebooted without a software image in either primary or secondary flash, the temporary flash image in RAM will be cleared and the switch will go down. To recover, refer to “Restoring a Flash Image” on page C-80 (in the “Troubleshooting” Appendix).

Syntax: copy flash flash <destination flash>

where: *destination flash* = **primary** or **secondary**:

For example, to copy the image in secondary flash to primary flash:

1. Verify that there is a valid flash image in the secondary flash location. The following figure indicates that a software image is present in secondary flash. (If you are unsure whether the image in secondary flash is valid, try booting from it before you proceed, by using **boot system flash secondary**.)

Figure 6-10. Example Indicating Two Different Software Versions in Primary and Secondary Flash

Execute the copy command as follows:

```
ProCurve(config)# copy flash flash primary
```

Erasing the Contents of Primary or Secondary Flash. This command deletes the software image file from the specified flash location.

Caution:

No Undo!

Before using this command in one flash image location (primary or secondary), ensure that you have a valid software file in the other flash image location (secondary or primary). If the switch has only one flash image loaded (in either primary or secondary flash) and you erase that image, then the switch does not have a software image stored in flash. In this case, if you do not reboot or power cycle the switch, you can recover by using xmodem or tftp to download another software image.

Syntax: erase flash < primary | secondary >

For example, to erase the software image in primary flash, do the following:

1. First verify that a usable flash image exists in secondary flash. The most reliable way to ensure this is to reboot the switch from the flash image you want to retain. For example, if you are planning to erase the primary image, then first reboot from the secondary image to verify that the secondary image is present and acceptable for your system:

```
ProCurve# boot system flash secondary
```

2. Then erase the software image in the selected flash (in this case, primary):

```
ProCurve# erase flash primary
The Primary OS Image will be deleted, continue [y/n]? _
```

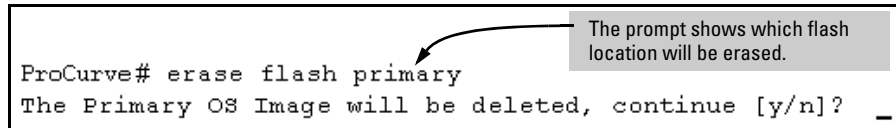


Figure 6-11. Example of Erase Flash Prompt

3. Type **y** at the prompt to complete the flash erase.
4. Use **show flash** to verify erasure of the selected software flash image

```
ProCurve# show flash
Compressed Primary Code size    = 0
Compressed Secondary Code size  = 2555802
Boot Rom Version:               E.05.04
Current Boot:                   Secondary
```

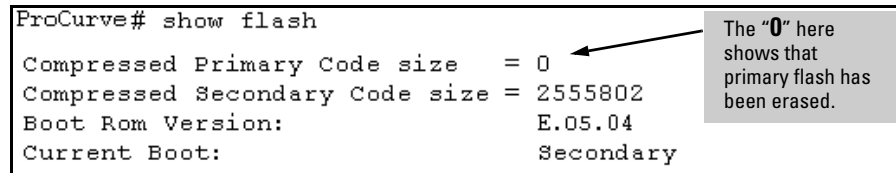


Figure 6-12. Example of Show Flash Listing After Erasing Primary Flash

In redundant management systems, this command will erase the selected flash in both the active and the standby management modules. If redundancy has been disabled or the standby module has failed selftest, this command only affects the active management module.

Rebooting the Switch

Operating Notes about Booting

Default Boot Source. The switch reboots from primary flash by default unless you specify the secondary flash by entering either the **boot system flash [primary | secondary]** or **boot set-default flash [primary | secondary]** command. Both the **boot** command and the **reload** command will reboot based on how these options have been selected.

Boot Attempts from an Empty Flash Location. In this case, the switch aborts the attempt and displays

```
Image does not exist
Operation aborted.
```

Interaction of Primary and Secondary Flash Images with the Current Configuration. The switch has one startup-config file (page 6-3), which it always uses for reboots, regardless of whether the reboot is from primary or secondary flash. Also, for rebooting purposes, it is not necessary for the software image and the startup-config file to support identical software fea-

tures. For example, suppose you have just downloaded a software upgrade that includes new features that are not supported in the software you used to create the current startup-config file. In this case, the software simply assigns factory-default values to the parameters controlling the new features. Similarly, If you create a startup-config file while using a version “Y” of the switch software, and then reboot the switch with an earlier software version “X” that does not include all of the features found in “Y”, the software simply ignores the parameters for any features that it does not support.

Scheduled Reload. If no parameters are entered after the **reload** command, an immediate reboot is executed. The **reload at** and **reload after** command information is not saved across reboots. If the switch is rebooted before a scheduled reload command is executed, the command is effectively cancelled. When entering a **reload at** or **reload after** command, a prompt will appear to confirm the command before it can be processed by the switch. For the **reload at** command, if *mm/dd/yy* are left blank, the current day is assumed.

The scheduled reload feature removes the requirement to physically reboot the switch at inconvenient times (for example, at 1:00 in the morning). Instead, a **reload at 1:00 mm/dd** command can be executed (where *mm/dd* is the date the switch is scheduled to reboot).

Boot and Reload Command Comparison

The switch offers reboot options through the **boot** and **reload** commands, plus the options inherent in a dual-flash image system. Generally, using **boot** provides more comprehensive self-testing; using **reload** gives you a faster reboot time.

Table 6-2. Comparing the Boot and Reload Commands

Actions	Included In Boot?	Included In Reload	Note
Save all configuration changes since the last boot or reload	Optional, with prompt	Optional with reload <cr>, when prompt displays. Not saved with reload at/after commands; No prompt is displayed.	Config changes saved to the startup-config file if "y" is selected (reload command).
Perform all system self-tests	Yes	No	The reload command provides a faster system reboot.
Choice of primary or secondary flash image	Yes	No—Uses the current flash image.	
Perform a scheduled reboot	No	Yes	Use the reload command with after/at parameters (see page 6-25 for details).

Setting the Default Flash

You can specify the default flash to boot from on the next boot by entering the **boot set-default flash** command.

Syntax: boot set-default flash [primary |secondary]

Upon booting, set the default flash for the next boot to primary or secondary.

```

ProCurve(config)# boot set-default flash secondary
ProCurve(config)# show flash
Image           Size(Bytes)    Date    Version    Build #
-----
Primary Image   : 7476770    03/15/07 K.12.XX    64
Secondary Image : 7476770    03/15/07 K.12.XX    64
Boot Rom Version: K.12.02
Default Boot    : Secondary

ProCurve(config)# boot
This management module will now reboot from secondary and will become
the standby module! You will need to use the other management module's
console interface. Do you want to continue [y/n]?

```

Figure 6-13. Example of boot set-default Command with Default Flash Set to Secondary (with a Redundant Management Module Present)

Booting from the Default Flash (Primary or Secondary)

The **boot** command boots the switch from the flash image that you are currently booted on, or the flash image that was set either by the **boot set-default** command or by the last executed **boot system flash <primary | secondary>** command. This command also executes the complete set of subsystem self-tests. You have the option of specifying a configuration file.

Syntax: boot [system [flash <primary | secondary>] [config FILENAME]

Reboots the switch from the flash that you are currently booted on (primary or secondary). You can select which image to boot from during the boot process itself. When using redundant management, the switch will failover to the standby management module.

Note: *This is changed from always booting from primary flash. You are prompted with a message which will indicate the flash being booted from.*

system: *Boots the switch. You can specify the flash image to boot from. When using redundant management, boots both the active and standby management modules.*

config: *You can optionally select a configuration file from which to boot.*

```
ProCurve(config)# boot
This management module will now reboot from primary image and will become
the standby module! You will need to use the other management module's
console interface. Do you want to continue [y/n]? y

Do you want to save current configuration [y/n]? n
```

Figure 6-14. Example of Boot Command (Default Primary Flash) with Redundant Management

In the above example, typing either a **y** or **n** at the second prompt initiates the reboot operation. (Entering **y** saves any configuration changes from the running-config file to the startup-config file; entering **n** discards them.)

```
ProCurve(config)# show flash
Image           Size(Bytes)   Date   Version  Build #
-----
Primary Image   : 7497114   03/29/07 K.12.XX   57
Secondary Image : 7497114   03/29/07 K.12.XX   57
Boot Rom Version: K.12.03
Default Boot    : Primary

ProCurve(config)# boot set-default flash secondary
This command changes the location of the default boot. This command will
change the default flash image to boot from secondary. Hereafter, 'reload'
'boot' commands will boot from secondary. Do you want to continue [y/n]? y

ProCurve(config)# boot
This management module will now reboot from secondary image and will become
the standby module! You will need to use the other management module's
console interface. Do you want to continue [y/n]? n
```

Figure 6-15. Example of Boot Command Booting from a Different Flash than the Current Flash (with Redundant Management Module Present)

Booting from a Specified Flash

This version of the boot command gives you the option of specifying whether to reboot from primary or secondary flash, and is the required command for rebooting from secondary flash. This option also executes the complete set of subsystem self-tests.

Syntax: boot system flash < primary | secondary >

For example, to reboot the switch from secondary flash when there are no pending configuration changes in the running-config file:

```
ProCurve(config)# boot system flash secondary
System will be rebooted from secondary image. Do you want to continue [y/n]?
```

Figure 6-16. Example of Boot Command with Secondary Flash Option

In the above example, typing either a **y** or **n** at the second prompt initiates the reboot operation.

Using the Fastboot feature. The **fastboot** command allows a boot sequence that skips the internal power-on self-tests, resulting in a faster boot time. When using redundant management and fastboot is enabled, it is saved to the standby management module when the config files are synchronized. Fastboot is used during the next bootup on either management module.

Syntax: [no] fastboot

Enables the fastboot option

*The **no** option disables the feature.*

Syntax: show fastboot

Shows the status of the fastboot feature, either enabled or disabled.

The fastboot command is shown below.

```
ProCurve(config)# fastboot
```

Using Reload

The **Reload** command reboots the switch from the flash image that you are currently booted on (primary or secondary) or the flash image that was set either by the **boot set-default** command or by the last executed **boot system flash <primary | secondary>** command. Because **reload** bypasses some subsystem self-tests, the switch reboots faster than if you use either of the **boot** command options. If you are using redundant management and redundancy is enabled, the switch will failover to the other management module.

Syntax: reload

For example, if you change the number of VLANs the switch supports, you must reboot the switch in order to implement the change. The **reload** command prompts you to save or discard the configuration changes.

```
ProCurve(config)# max-vlans 12
Command will take effect after saving configuration and reboot.

ProCurve(config)# reload
This command will cause a switchover to the other management module
which may not be running the same software image and configurations.
Do you want to continue [y/n]? y
```

Figure 6-17. Using Reload with with Redundant Management and Pending Configuration Changes

Scheduled Reload. Beginning with software release K.11.34, additional parameters have been added to the **reload** command to allow for a scheduled reboot of the switch via the CLI.

Syntax: [no] reload [after <[dd:]hh:]mm> | at <hh:mm[:ss]> [<mm/dd/[yy]yy>]]

Enables a scheduled warm reboot of the switch. The switch boots up with the same startup config file and using the same flash image as before the reload.

Caution: *When using redundant management, the **reload at/after** command causes a switchover at the scheduled time to the other management module, which may not be running the same software image or have the same configurations.*

Parameters include:

- **after:** *Schedules a warm reboot of the switch after a given amount of time has passed.*
- **at:** *Schedules a warm reboot of the switch at a given time.*

*The **no** form of the command removes a pending reboot request. For more details and examples, see below.*

The scheduled reload feature removes the requirement to physically reboot the switch at inconvenient times (for example, at 1:00 in the morning). Instead, a **reload at 1:00 mm/dd** command can be executed (where *mm/dd* is the date the switch is scheduled to reboot).

Note

Configuration changes are not saved with **reload at** or **reload after** commands. No prompt to save configuration file changes is displayed. See Table 6-2 on page 6-21.

Examples of scheduled **reload** commands:

- To schedule a reload in 15 minutes:
ProCurve# reload after 15
- To schedule a reload in 3 hours:
ProCurve# reload after 03:00
- To schedule a reload for the same time the following day:
ProCurve# reload after 01:00:00
- To schedule a reload for the same day at 12:05:
ProCurve# reload at 12:05
- To schedule a reload on some future date:
ProCurve# reload at 12:05 01/01/2008

```
ProCurve(config)# reload after 04:14:00
Reload scheduled in 4 days, 14 hours, 0 minutes
This command will cause a switchover at the scheduled time to the
other management module which may not be running the same software
image and configurations. Do you want to continue [y/n]?
```

Figure 6-18. An Example of the reload Command with a Redundant Management System

Multiple Configuration Files

Action	Page
Listing and Displaying Startup-Config Files	6-30
Changing or Overriding the Reboot Configuration Policy	6-31
Managing Startup-Config Files	
Renaming Startup-Config Files	6-34
Copying Startup-Config Files	6-34
Erasing Startup-Config Files	6-35
Effect of Using the Clear + Reset Buttons	6-37
Copying Startup-Config Files to or from a Remote Server	6-38

This method of operation means that you cannot preserve different startup-config files across a reboot without using remote storage.

The switch allows up to three startup-config files with options for selecting which startup-config file to use for:

- A fixed reboot policy using a specific startup-config file for a specific boot path (primary or secondary flash)
- Overriding the current reboot policy on a per-instance basis

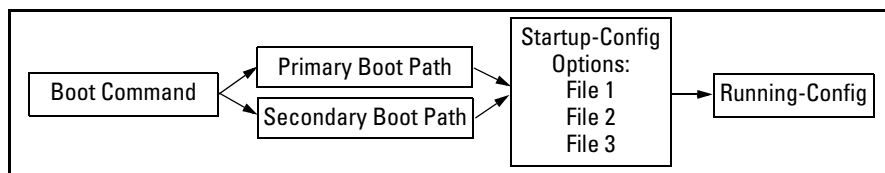


Figure 6-19. Optional Reboot Process

While you can still use remote storage for startup-config files, you can now maintain multiple startup-config files on the switch and choose which version to use for a reboot policy or an individual reboot.

This choice of which configuration file to use for the startup-config at reboot provides the following new options:

- The switch can reboot with different configuration options without having to exchange one configuration file for another from a remote storage location.
- Transitions from one software release to another can be performed while maintaining a separate configuration for the different software release versions.
- By setting a reboot policy using a known good configuration and then overriding the policy on a per-instance basis, you can test a new configuration with the provision that if an unattended reboot occurs, the switch will come up with the known, good configuration instead of repeating a reboot with a misconfiguration.

General Operation

Multiple Configuration Storage in the Switch. The switch uses three memory “slots”, with identity (**id**) numbers of **1**, **2**, and **3**.

```
ProCurve(config)# show config files
Configuration files:
  id | act pri sec | name
-----|-----|-----
  1 |   |   |   | oldConfig
  2 | *  *  *  * | workingConfig
  3 |   |   |   |
```

A startup-config file stored in a memory slot has a unique, changeable file name. The switches covered in this guide can use the startup-config in any of the memory slots (if the software version supports the configured features).

Boot Options. With multiple startup-config files in the switch you can specify a policy for the switch to use upon reboot. The options include:

- Use the designated startup-config file with either or both reboot paths (primary or secondary flash)
- Override the current reboot policy for one reboot instance by specifying a boot path (primary or secondary flash) and the startup-config file to use.

Changing the Startup-Config File. When the switch reboots, the startup-config file supplies the configuration for the running-config file the switch uses to operate. Making changes to the running-config file and then executing a **write-mem** command (or, in the Menu interface, the **Save** command) are written back to the startup-config file used at the last reboot. For example, suppose that a system administrator performs the following on a switch that has two startup-config files (**workingConfig** and **backupConfig**):

1. Reboot the switch through the Primary boot path using the startup-config file named **backupConfig**.
2. Use the CLI to make configuration changes in the running-config file, and then execute **write mem**.

The result is that the startup-config file used to reboot the switch is modified by the actions in step 2.

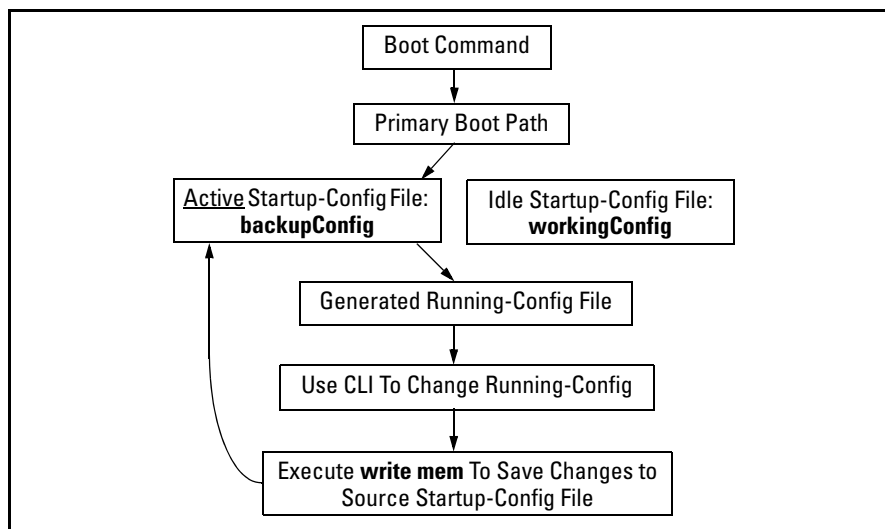


Figure 6-20. Example of Reboot Process and Making Changes to the Startup-Config File

Creating an Alternate Startup-Config File. There are two methods for creating a new configuration file:

- Copy an existing startup-config file to a new filename, then reboot the switch, make the desired changes to the running-config file, then execute **write memory**. (Refer to figure 6-6-20, above.)

- Erase the active startup-config file. This generates a new, default startup-config file that always results when the switch automatically reboots after deletion of the currently active startup-config file. (Refer to “Erasing a Startup-Config File” on page 6-35.)

Transitioning to Multiple Configuration Files

At the first reboot with a software release supporting multiple configuration, the switch:

- Assigns the filename **oldConfig** to the existing startup-config file (which is stored in memory slot 1).
- Saves a copy of the existing startup-config file in memory slot 2 with the filename **workingConfig**.
- Assigns the **workingConfig** file as the active configuration and the default configuration for all subsequent reboots using either primary or secondary flash.

```
ProCurve(config)# show config files
```

Configuration files:

id	act	pri	sec	name
1				oldConfig
2	*	*	*	workingConfig
3				

Figure 6-21. Switch Memory Assignments After the First Reboot from Software Supporting Multiple Configuration

In the above state, the switch always:

- Uses the **workingConfig** file to reboot

The commands described later in this section enable you to view the current multiple configuration status, manage multiple startup-config files, configure reboot policies, and override reboot policies on a per-instance basis.

Listing and Displaying Startup-Config Files

Command	Page
show config files	Below
show config < filename >	6-31

Viewing the Startup-Config File Status with Multiple Configuration Enabled

Rebooting the switch automatically enables the multiple configuration feature.

Syntax: show config files

This command displays the available startup-config files on the switch and the current use of each file.

id: Identifies the memory slot for each startup-config file available on the switch.

act: An asterisk (*) in this column indicates that the corresponding startup-config file is currently in use.

pri: An asterisk (*) in this column indicates that the corresponding startup-config file is currently assigned to the primary boot path.

sec: An asterisk (*) in this column indicates that the corresponding startup-config file is currently assigned to the secondary boot path.

name: Shows the filename for each listed startup-config file in the switch. Refer to “Renaming an Existing Startup-Config File” on page 6-34 for the command you can use to change existing startup-config filenames.

*In the default configuration, if the switch was shipped from the factory with software installed in both the primary and secondary boot paths, then one startup-config file named **config1** is used for both paths and is stored in memory slot 1. Memory slots 2 and 3 are empty in this default configuration.*

Displaying the Content of A Specific Startup-Config File

With Multiple Configuration enabled, the switch can have up to three startup-config files. Because the **show config** command always displays the content of the currently active startup-config file, the command extension shown below is needed to allow viewing the contents of any other startup-config files stored in the switch.

Syntax: show config < filename >

*This command displays the content of the specified startup-config file in the same way that the **show config** command displays the content of the default (currently active) startup-config file.*

Changing or Overriding the Reboot Configuration Policy

Command	Page
startup-default [primary secondary] config < filename >	Below
boot system flash < primary secondary > config < filename >	6-33

You can boot the switch using any available startup-config file.

Changing the Reboot Configuration Policy. For a given reboot, the switch automatically reboots from the startup-config file assigned to the flash location (primary or secondary) being used for the current reboot. For example, when you first download a software version that supports multiple configuration files and boot from the flash location of this version, the switch copies the existing startup-config file (named **oldConfig**) into memory slot 2, renames this file to **workingConfig**, and assigns **workingConfig** as:

- The active configuration file
- The configuration file to use when booting from either primary or secondary flash.

In this case, the switch is configured to automatically use the **workingConfig** file in memory slot 2 for all reboots.

You can use the following command to change the current policy so that the switch automatically boots using a different startup-config file.

Syntax: startup-default [primary | secondary] config < filename >

Specifies a boot configuration policy option:

[primary | secondary] config < filename >: Designates the startup-config file to use in a reboot with the software version stored in a specific flash location. Use this option to change the reboot policy for either primary or secondary flash, or both.

config < filename >: Designates the startup-config file to use for all reboots, regardless of the flash version used. Use this option when you want to automatically use the same startup-config file for all reboots, regardless of the flash source used.

For redundant management systems, this command affects both the active management module and the standby management module. The config file is copied immediately to the standby management module and becomes the default on that module when the next bootup occurs, unless redundancy is disabled or the standby module has failed selftest.

Note: To override the current reboot configuration policy for a single reboot instance, use the **boot system flash** command with the options described under “Overriding the Default Reboot Configuration Policy” on page 6-33.

For example, suppose:

- Software release “A” is stored in primary flash and a later software release is stored in secondary flash.
- The system operator is using memory slot 1 for a reliable, minimal configuration (named **minconfig**) for the software version in the primary flash, and slot 2 for a modified startup-config file (named **newconfig**) that includes untested changes for improved network operation with the software version in secondary flash.

The operator wants to ensure that in case of a need to reboot by pressing the Reset button, or if a power failure occurs, the switch will automatically reboot with the minimal startup-config file in memory slot 1. Since a reboot due to pressing the Reset button or to a power cycle always uses the software version in primary flash, the operator needs to configure the switch to always boot from primary flash with the startup-config file named **minconfig** (in memory slot 1). Also, whenever the switch boots from secondary flash, the operator also wants the startup-config named **newconfig** to be used. The following two commands configure the desired behavior.


```
ProCurve(config) # startup-default pri config minconfig  
ProCurve(config) # startup-default sec config newconfig.
```

Overriding the Default Reboot Configuration Policy. This command provides a method for manually rebooting with a specific startup-config file other than the file specified in the default reboot configuration policy.

Syntax: boot system flash < primary | secondary > config < filename >

Specifies the name of the startup-config file to apply for the immediate boot instance only. This command overrides the current reboot policy.

Using Reload To Reboot From the Current Flash Image and Startup-Config File.

Syntax: reload

*This command boots the switch from the currently active flash image and startup-config file. Because **reload** bypasses some subsystem self-tests, the switch boots faster than if you use a **boot** command.*

Note: To identify the currently active startup-config file, use the **show config files** command.

Managing Startup-Config Files in the Switch

Command	Page
rename config < current-filename > < newname-str >	6-34
copy config < source-filename > config < dest-filename >	6-34
erase config < filename > startup-config	6-35
Erase startup-config using the front-panel Clear + Reset Buttons	6-37

Renaming an Existing Startup-Config File

Syntax: `rename config < current-filename > < newname-str >`

This command changes the name of an existing startup-config file. A file name can include up to 63, alphanumeric characters. Blanks are allowed in a file name enclosed in quotes (“ ” or ‘ ’). (File names are not case-sensitive.)

For redundant management systems, renaming a config file affects both the active management module and the standby management module, unless redundancy is disabled or the standby module failed selftest.

Creating a New Startup-Config File

The switch allows up to three startup-config files. You can create a new startup-config file if there is an empty memory slot or if you want to replace one startup-config file with another.

Syntax: `copy config < source-filename > config < target-filename >`

This command makes a local copy of an existing startup-config file by copying the contents of an existing startup-config file in one memory slot to a new startup-config file in another, empty memory slot. This enables you to use a separate configuration file to experiment with configuration changes, while preserving the source file unchanged. It also simplifies a transition from one software version to another by enabling you to preserve the startup-config file for the earlier software version while creating a separate startup-config file for the later software version. With two such versions in place, you can easily reboot the switch with the correct startup-config file for either software version.

- *If the destination startup-config file already exists, it is overwritten by the content of the source startup-config file.*
- *If the destination startup-config file does not already exist, it will be created in the first empty configuration memory slot on the switch.*
- *If the destination startup-config file does not already exist, but there are no empty configuration memory slots on the switch, then a new startup-config file is not created and instead, the CLI displays the following error message:*

Unable to copy configuration to “< target-filename >” .

For example, suppose both primary and secondary flash memory contain software release “A” and use a startup-config file named **config1**:

```
ProCurve(config)# show config files

Configuration files:

id | act pri sec | name
-----|-----|-----
1  | *  *  *  | config1
2  |
3  |
```

Figure 6-22. Example of Using One Startup-Config File for Both Primary and Secondary Flash

If you wanted to experiment with configuration changes to the software version in secondary flash, you could create and assign a separate startup-config file for this purpose.

```
ProCurve(config)# copy config config1 config config2
ProCurve(config)# startup-default secondary config config2
ProCurve(config)# show config files

Configuration files:

id | act pri sec | name
-----|-----|-----
1  | *  *  *  | config1
2  |                | config2
3  |
```

The first two commands copy the **config1** startup-config file to **config2**, and then make **config2** the default startup-config file for booting from secondary flash.

Figure 6-23. Example of Creating and Assigning a New Startup-Config File

Note

You can also generate a new startup-config file by booting the switch from a flash memory location from which you have erased the currently assigned startup-config file. Refer to “Erasing a Startup-Config File” in the next section.

Erasing a Startup-Config File

You can erase any of the startup-config files in the switch’s memory slots. In some cases, erasing a file causes the switch to generate a new, default-configuration file for the affected memory slot.

In a redundant management system, this command erases the config or startup config file on both the active and the standby management modules as long as redundancy has not been disabled. If the standby management module is not in standby mode or has failed selftest, the config or startup config file is not erased.

Syntax: erase < config < filename >> | startup-config >

config < filename >: *This option erases the specified startup-config file. If the specified file is not the currently active startup-config file, then the file is simply deleted from the memory slot it occupies. If the specified file is the currently active startup-config file, then the switch creates a new, default startup-config file with the same name as the erased file, and boots using this file. (This new startup-config file contains only the default configuration for the software version used in the reboot.)*

Note: *Where a file is assigned to either the primary or the secondary flash, but is not the currently active startup-config file, erasing the file does not remove the flash assignment from the memory slot for that file. Thus, if the switch boots using a flash location that does not have an assigned startup-config, then the switch creates a new, default startup-config file and uses this file in the reboot. (This new startup-config file contains only the default configuration for the software version used in the reboot.) Executing **write memory** after the reboot causes a switch-generated filename of **configx** to appear in the **show config files** display for the new file, where **x** corresponds to the memory slot number.*

startup-config: *This option erases the currently active startup-config file and reboots the switch from the currently active flash memory location. The erased startup-config file is replaced with a new startup-config file. The new file has the same filename as the erased file, but contains only the default configuration for the software version in the flash location (primary or secondary) used for the reboot. For example, suppose the last reboot was from primary flash using a configuration file named **minconfig**. Executing **erase startup-config** replaces the current content of **minconfig** with a default configuration and reboots the switch from primary flash.*

Figure 6-24 illustrates using **erase config < filename >** to remove a startup-config file.

```

ProCurve(config)# show config files

Configuration files:

id | act pri sec | name
-----
 1 | *   *   | minconfig
 2 |       *   | config2
 3 |       | config3

ProCurve(config)# erase config config3
ProCurve(config)# show config files

Configuration files:

id | act pri sec | name
-----
 1 | *   *   | minconfig
 2 |       *   | config2
 3 |       |

```

Figure 6-24. Example of Erasing a Non-Active Startup-Config File

With the same memory configuration as is shown in the bottom portion of figure 6-24, executing **erase startup-config** boots the switch from primary flash, resulting in a new file named **minconfig** in the same memory slot. The new file contains the default configuration for the software version currently in primary flash.

Using the Clear + Reset Button Combination To Reset the Switch to Its Default Configuration

The Clear + Reset button combination described in the *Installation and Getting Started Guide* produces these results. That is, when you press the Clear + Reset button combination, the switch:

- Overwrites the content of the startup-config file currently in memory slot 1 with the default configuration for the software version in primary flash, and renames this file to **config1**.
- Erases any other startup-config files currently in memory.
- Configures the new file in memory slot 1 as the default for both primary and secondary flash locations (regardless of the software version currently in secondary flash).
- Boots the switch from primary flash using the new startup-config file.

```
ProCurve Switch 5304XL# sho config files
```

id	act	pri	sec	name
1	*	*	*	config1
2				
3				

Pressing Clear + Reset:

- Replaces all startup-config files with a single file named **config1** that contains the default configuration for the software version in primary flash.
- Resets the Active, Primary, and Secondary assignments as shown here.

Figure 6-25. Example of Clear + Reset Result

Transferring Startup-Config Files To or From a Remote Server

Command	Page
copy config < src-file > tftp < ip-addr > < remote-file > < pc unix >	below
copy tftp config < dest-file > < ip-addr > < remote-file > < pc unix >	below
copy config < src-file > xmodem < pc unix >	6-39
copy xmodem config < dest-file > < pc unix >	6-40

TFTP: Copying a Configuration File to a Remote Host

Syntax: copy config < src-file > tftp < ip-addr > < remote-file > < pc | unix >

This is an addition to the copy tftp command options. Use this command to upload a configuration file from the switch to a TFTP server.

For more on using TFTP to copy a file to a remote server, refer to “TFTP: Copying a Configuration File to a Remote Host” on page A-25.

For example, the following command copies a startup-config file named **test-01** from the switch to a (UNIX) TFTP server at IP address 10.10.28.14:

```
ProCurve(config)# copy config test-01 tftp 10.10.28.14
test-01.txt unix
```

TFTP: Copying a Configuration File from a Remote Host

Syntax: `copy tftp config < dest-file > < ip-addr > < remote-file > < pc | unix >`

This is an addition to the `copy tftp` command options. Use this command to download a configuration file from a TFTP server to the switch.

Note: This command requires an empty memory slot in the switch. If there are no empty memory slots, the CLI displays the following message:

Unable to copy configuration to "< filename >".

For more on using TFTP to copy a file from a remote host, refer to “TFTP: Copying a Configuration File from a Remote Host” on page A-25.

For example, the following command copies a startup-config file named **test-01.txt** from a (UNIX) TFTP server at IP address 10.10.28.14 to the first empty memory slot in the switch:

```
ProCurve(config)# copy tftp config test-01 10.10.28.14  
test-01.txt unix
```

Xmodem: Copying a Configuration File to a Serially Connected Host

Syntax: `copy config < filename > xmodem < pc | unix >`

This is an addition to the `copy < config > xmodem` command options. Use this command to upload a configuration file from the switch to an Xmodem host.

For more on using Xmodem to copy a file to a serially connected host, refer to “Xmodem: Copying a Configuration File to a Serially Connected PC or UNIX Workstation” on page A-26.

Xmodem: Copying a Configuration from a Serially Connected Host

Syntax: copy xmodem config < dest-file > < pc | unix >

*This is an addition to the **copy xmodem** command options. Use this command to download a configuration file from an Xmodem host to the switch.*

For more on using Xmodem to copy a file from a serially connected host, refer to “Xmodem: Copying a Configuration File from a Serially Connected PC or UNIX Workstation” on page A-26.

Operating Notes for Multiple Configuration Files

- SFTP/SCP: The configuration files are available for sftp/scp transfer as **/cfg/< filename >**.