CHAPTER 25

Modifying the Switch Boot Configuration

This chapter describes how to modify the switch boot configuration on the Catalyst 6500 series switches, including the BOOT environment variable, the CONFIG_FILE environment variable, and the configuration register.

Note For complete syntax and usage information for the commands that are used in this chapter, refer to the *Catalyst 6500 Series Switch Command Reference* publication.

This chapter consists of these sections:

Understanding How the Switch Boot Configuration Works

These sections describe how the boot configuration works:

Understanding the Boot Process

The boot process involves two software images: ROM monitor and supervisor engine system code. When you power up or reset the switch, the ROM-monitor code is executed. Depending on the nonvolatile RAM (NVRAM) configuration, the switch either stays in ROM-monitor mode or loads the supervisor engine system code.

Two user-configurable parameters determine how the switch boots: the configuration register and the BOOT environment variable. The configuration register is described in the "Understanding the Configuration Register" section on page 25-2. The BOOT environment variable is described in the "Understanding the BOOT Environment Variable" section on page 25-3.

Understanding the ROM Monitor

The ROM-monitor code executes upon switch power up, reset, or when a fatal exception occurs. The system enters ROM-monitor mode if the switch does not find a valid system image, if the NVRAM configuration is corrupted, or if the configuration register is set to enter ROM-monitor mode. From ROM-monitor mode, you can manually load a system image from flash memory, from a network server file, or from bootflash.

You can enter ROM-monitor mode by restarting the switch and then pressing the **Break** key during the first 60 seconds of startup. If you are connected through a terminal server, you can escape to the Telnet prompt and enter the **send break** command to enter ROM-monitor mode.

Note The **Break** key is always enabled for 60 seconds after rebooting the system, regardless of whether the configuration-register setting has the **Break** key disabled.

The following functionality is built into the ROM monitor:

- Power-on confidence test
- Hardware initialization
- Boot capability (allows manual boot and autoboot)
- Debug utility and crash analysis
- Monitor call interface (EMT calls—the ROM monitor provides information and some functionality to the running system images through EMT calls)
- File system (the ROM monitor knows the simple file system and supports the newly developed file system through the dynamic linked file system library [MONLIB])
- Exception handling

Understanding the Configuration Register

The configuration register determines whether the switch loads an operating system image and where the system image is stored. The configuration register boot field determines if and how the ROM monitor loads a supervisor engine system image at startup. You can modify the boot field to force the switch to boot a particular system image at startup instead of using the default system image.

The lowest four bits (bits 3, 2, 1, and 0) of the 16-bit configuration register form the boot field. The default boot field value is 0x10F. The possible configuration register boot field settings are as follows:

- When the boot field equals 0000, the switch does not load a system image. Instead, it enters ROM-monitor mode from which you can enter ROM-monitor commands to load a system image manually.
- When the boot field equals 0001, the switch loads the first valid system image found in onboard flash memory.
- When the boot field equals a value between 0010 and 1111, the switch loads the system image that is specified by the **boot system** commands in the NVRAM configuration. It attempts to boot the image in the order in which you entered the **boot system** commands. If it cannot boot any image in the BOOT environment variable list, the switch remains in ROM-monitor mode. The exact booting sequence is defined by the ROM monitor.

The other bits in the configuration register function as follows when set:

- Bit 5 (0x0020)—Enables CONFIG_FILE recurrence.
- Bit 6 (0x0040)—Causes system software to clear NVRAM contents.
- Bit 7 (0x0080)—Enables OEM bit (not used).
- Bit 8 (0x0100)—Disables break.
- Bit 9 (0x0200)—Uses secondary bootstrap (not used by the ROM monitor).
- Bit 10 (0x0400)—Provides IP broadcast with all zeros (not used).
- Bits 11/12 (0x0800/0x1000)—Provide console line speed: 0/0=9600, 0/1=1200, 1/0=4800, 1/1=2400 (default is 9600).
- Bit 13 (0x2000)—Boots default flash software if network boot fails (not used).
- Bit 14 (0x4000)—IP broadcasts do not have network numbers (not used).
- Bit 15 (0x8000)—Enables diagnostic messages and ignores NVRAM contents (not used).

Understanding the BOOT Environment Variable

The BOOT environment variable specifies a list of image files on the various devices from which the switch can boot at startup.

You can add several images to the BOOT environment variable to provide a fail-safe boot configuration. If the first file fails to boot the switch, subsequent images that are specified in the BOOT environment variable are tried until the switch boots or there are no additional images to attempt to boot. If there is no valid image to boot, the system enters ROM-monitor mode where you can manually specify an image to boot.

The system stores and executes images in the order in which you added them to the BOOT environment variable. If you want to change the order in which the images are tried at startup, you can either prepend and clear images from the BOOT environment variable to attain the desired order or you can clear the entire BOOT environment variable and then redefine the list in the desired order.

Understanding the CONFIG_FILE Environment Variable

You can use the CONFIG_FILE environment variable to specify a list of configuration files (auto-config files) on the various devices to use to configure the switch at startup. You can specify the following functions:

- Nonrecurring—When you add a list of configuration files to the CONFIG_FILE environment variable, the next time that the switch is restarted, the system erases the configuration in NVRAM and uses the specified files to configure the switch. The CONFIG_FILE environment variable is cleared before the switch is configured. Nonrecurring is the default setting.
- Recurring—When you add a list of configuration files to the CONFIG_FILE environment variable, the list is stored indefinitely in NVRAM. Each time that the switch is restarted, the system erases the configuration in NVRAM and configures the switch using the configuration files that are specified. The CONFIG_FILE environment variable is not cleared.

  For information on specifying recurrence or nonrecurrence, see the "Setting CONFIG_FILE Recurrence" section on page 25-7.
- Overwrite—When you add a list of configuration files to the CONFIG_FILE environment variable, overwriting means that the NVRAM configuration will be cleared before executing the configuration files. Overwrite is the default setting.
- Append—Append means that the configuration files will be executed without first clearing NVRAM.

  For information on specifying overwriting or appending, see the "Setting CONFIG_FILE Overwrite" section on page 25-8.
- Sync enable—Enables synchronization to force the configuration files to synchronize automatically to the standby supervisor engine. The file(s) are kept consistent with what is on the active supervisor engine.
- Sync disable—Disables synchronization.

  For information on specifying synchronization, see the "Setting CONFIG_FILE Synchronization" section on page 25-8.

Tip You can alter the CONFIG_FILE environment variable or change its other properties by using the commands in the configuration files that configure the switch at startup.

You can add multiple configuration files to the CONFIG_FILE environment variable. The specified files can be any valid configuration file that is stored on a local flash device (bootflash: or slot0:).

When the switch boots up, if any of the files that are specified in the CONFIG_FILE environment variable are valid configuration files, the configuration in NVRAM is erased and the system uses the specified configuration file to configure the switch. If multiple valid configuration files are specified, each configuration file is executed in the order in which it appears in the CONFIG_FILE environment variable.

If any specified file is not a valid configuration file, the entry is skipped and subsequent files are tried until there are no additional, specified images. If no valid configuration file is specified, the system retains the last configuration that is stored in NVRAM.

Default Switch Boot Configuration

Table 25-1 shows the default switch boot configuration.

Table 25-1 Default Switch Boot Configuration

Feature	Default Configuration
Configuration register value	0x10f
Boot method	System boots from the image that is specified in the BOOT environment variable
ROM-monitor console port baud rate	9600 baud
ignore-config parameter	Disabled
BOOT environment variable	Empty
CONFIG_FILE environment variable	slot0:switch.cfg
CONFIG_FILE recurrence configuration register parameter	Nonrecurring
CONFIG_FILE overwrite configuration register parameter	Overwrite
CONFIG_FILE synchronization configuration register parameter	Synchronization disabled

Setting the Configuration Register

Note The configuration register settings are not copied automatically to a redundant supervisor engine. You must set the configuration register separately for each supervisor engine in the switch.

These sections describe how to modify the configuration register:

- Setting the Boot Field in the Configuration Register, page 25-6
- Setting the ROM-Monitor Console-Port Baud Rate, page 25-6
- Setting CONFIG_FILE Recurrence, page 25-7
- Setting CONFIG_FILE Overwrite, page 25-8
- Setting CONFIG_FILE Synchronization, page 25-8
- Setting the Switch to Ignore the NVRAM Configuration, page 25-9
- Setting the Configuration Register Value, page 25-10

Setting the Boot Field in the Configuration Register

You can determine the boot method that the switch will use at the next startup by setting the boot field in the configuration register. This command affects only the configuration register bits that control the boot field and leaves the remaining bits unaltered. The following boot methods are supported:

- ROM monitor—Enter the **rommon** keyword to force the switch to remain in ROM-monitor mode at startup.
- Bootflash—Enter the **bootflash** keyword to cause the switch to boot from the first image that is stored in the onboard flash memory.
- System—Enter the **system** keyword to boot from the image that is specified in the BOOT environment variable (the default).

Note We recommend that you use only the **rommon** and **system** keywords with the **set boot config-register boot** command.

To set the configuration register boot field, perform this task in privileged mode:

Task	Command
Set the boot field in the configuration register.	**set boot config-register boot** {**rommon** | **bootflash** | **system**} [*mod*]

This example shows how to set the boot field in the configuration register:

```
Console> (enable) set boot config-register boot rommon
Configuration register is 0x0
ignore-config: disabled
auto-config: non-recurring
console baud: 9600
boot: the ROM monitor
Console> (enable)
```

Setting the ROM-Monitor Console-Port Baud Rate

You can set the console-port baud rate that is used by the ROM monitor. The new baud rate is used the next time that the switch is restarted. This command affects only the configuration register bits that control the baud rate and leaves the remaining bits unaltered.

Note The baud rate that is specified in the configuration register is used by the ROM monitor only and is different from the baud rate that is specified by the **set system baud** command.

To set the ROM-monitor console-port baud rate in the configuration register, perform this task in privileged mode:

Task	Command
Set the ROM-monitor console-port baud rate in the configuration register.	**set boot config-register baud** {**1200** | **2400** | **4800** | **9600**} [*mod*]

This example shows how to set the ROM-monitor console-port baud rate in the configuration register to 2400:

```
Console> (enable) set boot config-register baud 2400
Configuration register is 0x1800
ignore-config: disabled
auto-config: non-recurring
console baud: 2400
boot: the ROM monitor
Console> (enable)
```

Setting CONFIG_FILE Recurrence

By default, when you set the CONFIG_FILE environment variable, the list of configuration files to use at startup is retained only until the next time that the switch is restarted.

You can cause the system software to retain the CONFIG_FILE environment variable settings indefinitely so that each time that the switch is restarted, the specified configuration files are used to configure the switch.

This command affects only the configuration register bit that controls whether the CONFIG_FILE environment variable settings are recurring or nonrecurring. The remaining configuration register bits are unaltered.

Caution With the CONFIG_FILE environment variable set to **recurring**, the current configuration in NVRAM is erased each time that the switch is restarted and the switch is configured using the specified configuration files. With the CONFIG_FILE environment variable set to **non-recurring**, the current configuration in NVRAM is erased at the next restart and the switch is configured using the specified configuration files. The NVRAM configuration is retained after subsequent restarts (unless you again set the CONFIG_FILE variable).

To set the switch to retain the current CONFIG_FILE environment variable indefinitely, perform this task in privileged mode:

Task	Command
Set the switch to retain the current CONFIG_FILE environment variable indefinitely.	**set boot config-register auto-config** {**recurring** | **non-recurring**}

This example shows how to set the switch to retain the current CONFIG_FILE environment variable indefinitely:

```
Console> (enable) set boot config-register auto-config recurring
Configuration register is 0x1820
ignore-config: disabled
auto-config: recurring, overwrite, sync disabled
console baud: 2400
boot: the ROM monitor
Console> (enable)
```

Setting CONFIG_FILE Overwrite

This command allows you to specify if the auto-config file should be used to overwrite the NVRAM configuration or if the file configuration should be appended to what is currently in NVRAM. Overwriting means that the NVRAM configuration will be cleared before executing the auto-config file; appending means that the auto-config file will be executed without first clearing NVRAM. The default is **overwrite**.

To specify if the auto-config file should be used to overwrite the NVRAM configuration or if the file configuration should be appended to what is currently in NVRAM, perform this task in privileged mode:

Task	Command
Specify if the auto-config file should be used to overwrite the NVRAM configuration or if the file configuration should be appended to what is currently in NVRAM.	**set boot config-register auto-config** {**overwrite** | **append**}

This example shows how to specify that the auto-config file is used to overwrite the NVRAM configuration:

```
Console> (enable) set boot config-register auto-config overwrite
Configuration register is 0x12F
ignore-config: disabled
auto-config: recurring, overwrite, sync disabled
console baud: 9600
boot: image specified by the boot system commands
Console> (enable)
```

This example shows how to specify that the auto-config file is appended to what is currently in NVRAM:

```
Console> (enable) set boot config-register auto-config append
Configuration register is 0x12F
ignore-config: disabled
auto-config: recurring, append, sync disabled
console baud: 9600
boot: image specified by the boot system commands
Console> (enable)
```

Setting CONFIG_FILE Synchronization

The **set boot config-register auto-config sync** command allows you to enable synchronization to force the auto-config file(s) to synchronize automatically to the standby supervisor engine. The file(s) are kept consistent with what is on the active supervisor engine. The default is **disabled**. These events can trigger a synchronization check and a synchronization (if necessary):

- Changing the auto-config file(s) on either supervisor engine (if the file is deleted on the active supervisor engine, it is also deleted on the standby supervisor engine)
- Changing the boot string CONFIG_FILE variable setting
- Inserting a new supervisor engine
- System startup

The CONFIG_FILE variable from the active supervisor engine is made identical on the standby supervisor engine. Each auto-config file on the active supervisor engine is compared against each corresponding auto-config file on the standby supervisor engine. Two files are considered identical if their lengths and cyclic redundancy check (CRC) are the same. If a file on the standby supervisor engine is not identical to the file on the active supervisor engine, a new file is generated on the standby supervisor engine with the name of the file on the active supervisor engine. If a file with that name already exists on the standby supervisor engine, it is overwritten.

To enable or disable synchronization, perform this task in privileged mode:

Task	Command
Specify if synchronization should be enabled or disabled.	**set boot config-register auto-config sync** {**enable** | **disable**}

This example shows how to enable synchronization:

```
Console> (enable) set boot config-register auto-config sync enable
Configuration register is 0x12F
ignore-config: disabled
auto-config: recurring, append, sync enabled
console baud: 9600
boot: image specified by the boot system commands
Console> (enable)
```

This example shows how to disable synchronization:

```
Console> (enable) set boot config-register auto-config sync disable
Configuration register is 0x12F
ignore-config: disabled
auto-config: recurring, append, sync disabled
console baud: 9600
boot: image specified by the boot system commands
Console> (enable)
```

Setting the Switch to Ignore the NVRAM Configuration

You can cause the system software to ignore the configuration information that is stored in NVRAM the next time that the switch is restarted. The **set boot config-register ignore-config enable** command affects only the configuration register bits that control whether the switch ignores the NVRAM configuration and leaves the remaining bits unaltered. This command affects the next system restart only.

Caution Enabling the **ignore-config** parameter is the same as entering the **clear config all** command; that is, it clears the entire configuration that is stored in NVRAM the next time that the switch is restarted.

To set the switch to ignore the NVRAM configuration at the next startup, perform this task in privileged mode:

Task	Command
Set the switch to ignore the contents of NVRAM at startup.	**set boot config-register ignore-config enable**

This example shows how to set the switch to ignore the NVRAM configuration at the next startup:

```
Console> (enable) set boot config-register ignore-config enable
Configuration register is 0x1860
ignore-config: enabled
auto-config: recurring
console baud: 2400
boot: the ROM monitor
Console> (enable)
```

Setting the Configuration Register Value

To set the configuration register value, perform this task in privileged mode:

Task	Command
Set the configuration register.	**set boot config-register 0x***value* [*mod*]

This example shows how to set the configuration register value to 0x90f:

```
Console> (enable) set boot config-register 0x90f
Configuration register is 0x90f
ignore-config: disabled
auto-config: non-recurring
console baud: 4800
boot: image specified by the boot system commands
Console> (enable)
```

Setting the BOOT Environment Variable

Note The BOOT environment variable settings are not copied automatically to a redundant supervisor engine (if present). You must set the BOOT variable separately for each supervisor engine in the switch.

These sections describe how to modify the BOOT environment variable:

- Setting the BOOT Environment Variable, page 25-10
- Clearing the BOOT Environment Variable Settings, page 25-11

Setting the BOOT Environment Variable

To set the BOOT environment variable, perform this task in privileged mode:

Task	Command
Set the BOOT environment variable.	**set boot system flash** *device:*[*filename*] [**prepend**] [*mod*]

This example shows how to set the BOOT environment variable:

```
Console> (enable) set boot system flash bootflash:cat6000-sup.5-5-1.bin
BOOT variable = bootflash:cat6000-sup.5-5-1.bin,1;
Console> (enable) set boot system flash bootflash:cat6000-sup.4-5-2.bin
BOOT variable = bootflash:cat6000-sup.5-1-1.bin,1;bootflash:cat6000-sup.4-5-2.
bin,1;
Console> (enable) set boot system flash bootflash:cat6000-sup.5-2-1.bin prepend
BOOT variable = bootflash:cat6000-sup.5-2-1.bin,1;bootflash:cat6000-sup.5-5-1.
bin,1;bootflash:cat6000-sup.4-5-2.bin,1;
Console> (enable)
```

Clearing the BOOT Environment Variable Settings

To clear the entries from the BOOT environment variable, perform one of these tasks in privileged mode:

Task	Command
Clear a specific image from the BOOT environment variable.	**clear boot system flash** *device:*[*filename*] [*mod*]
Clear the entire BOOT environment variable.	**clear boot system all** [*mod*]

This example shows how to clear a specific entry from the BOOT environment variable:

```
Console> (enable) clear boot system flash bootflash:cat6000-sup.5-1-1.bin
BOOT variable = bootflash:cat6000-sup.5-2-1.bin,1;bootflash:cat6000-sup.4-5-2.
bin,1;
Console> (enable)
```

This example shows how to clear the entire BOOT environment variable:

```
Console> (enable) clear boot system all
BOOT variable =
Console> (enable)
```

Setting the CONFIG_FILE Environment Variable

These sections describe how to modify the CONFIG_FILE environment variable:

Setting the CONFIG_FILE Environment Variable

You can specify multiple configuration files with the **set boot auto-config** command by separating them with a semicolon (;). You must specify both the device name and the filename for each configuration file.

Note You cannot prepend or append the configuration files to the CONFIG_FILE environment variable. Entering the **set boot auto-config** command erases any list of configuration files that were previously specified using the **set boot auto-config** command.

To set the CONFIG_FILE environment variable, perform this task in privileged mode:

Task	Command
Set the CONFIG_FILE environment variable.	**set boot auto-config** *device:filename*[**;***device:filename*...]

This example shows how to set the CONFIG_FILE environment variable:

```
Console> (enable) set boot auto-config bootflash:generic.cfg;bootflash:6509_1_noc.cfg
CONFIG_FILE variable = bootflash:generic.cfg;bootflash:6509_1_noc.cfg
WARNING: nvram configuration may be lost during next bootup,
         and re-configured using the file(s) specified.
Console> (enable)
```

Clearing the CONFIG_FILE Environment Variable Settings

To clear the entries from the CONFIG_FILE environment variable, perform this task in privileged mode:

Task	Command
Clear the entries in the CONFIG_FILE environment variable.	**clear boot auto-config**

This example shows how to clear the entries in the CONFIG_FILE environment variable:

```
Console> (enable) clear boot auto-config
CONFIG_FILE variable =
Console> (enable)
```

Displaying the Switch Boot Configuration

To display the current configuration register, the BOOT environment variable, and the CONFIG_FILE environment variable settings, perform this task:

Task	Command
Display the current configuration register, the BOOT environment variable, and the CONFIG_FILE environment variable settings.	**show boot** [*mod*]

This example shows how to display the current configuration register, the BOOT environment variable, and the CONFIG_FILE environment variable settings:

```
Console> (enable) show boot
BOOT variable = bootflash:cat6000-sup.5-2-1.bin,1;
CONFIG_FILE variable = bootflash:generic.cfg;bootflash:6509_1_noc.cfg

Configuration register is 0x12f
ignore-config: disabled
auto-config: recurring
console baud: 9600
boot: image specified by the boot system commands

Console> (enable)
```