

Cool IOS Commands

Michael J. Martin

While it may be hard to believe, many administrators only think about the router's Internetwork Operating System (IOS) and configuration after disaster strikes. Corrupted IOS versions, lost router configurations, or being locked out of a router because a password was changed or lost -- these things happen to other administrators, not you.

What about IOS upgrades, how do you handle those? Each major release of the Cisco IOS (a major release is a mainline version number (i.e., 10, 11, 12) followed by a release version number (i.e., X.1, X.2, etc) has a three-year lifespan. It usually takes about a year before a release moves from Limited Deployment (LD) to General Deployment (GD) and then there are the numerous Early Deployment (ED) releases that provide IOS enhancements and bug fixes. Controlling how the router gets its operating system and configuration information is often taken for granted, but it shouldn't be.

This article focuses on working with the IOS image and router configuration files, reviewing the router's default bootstrap behavior, and looks at implementing alternative IOS loading and configuration loading options. We also include an overview of Internetworking File System (IFS) file management tools.

The IOS, Router Memory and Datasets

All Cisco routers (most Cisco products, for that matter) run the IOS. The IOS is a kernel based operating system optimized for network communications. In terms of actual media, the "IOS" is a single self-decompressing binary image that is loaded from a local or remote storage point. On Cisco routers, for the most part, there are two types of local storage media: FLASH, which can be either Linear FLASH or ATA-PCMCIA Compact Flash and NVRAM, a small chunk of RAM between 32 K and 128 K in size. This is kept alive by a battery in the system board so the contents are not lost. IOS images are stored on FLASH and the system configuration is stored on NVRAM.

Being software, the IOS must be loaded into the router's system RAM memory in order for the router to function. In fact, pretty much everything happens in RAM on Cisco routers. That's because of speed. RAM operates at the same bus speed as the processor and does not have the added overhead of needing to be fetched from physical magnetic media through repeated "reads" from a mechanical drive head. In addition to the IOS, the router's RAM also contains a number of key datasets. Routing tables, switching caches, ARP tables, packet and queuing buffers all reside in RAM. So the amount of installed RAM a router has partly determines overall performance. Simply put, if the router does not possess enough RAM to store the IOS and the various tables and caches needed for the router to function, it crashes. ***It is critical that you follow Cisco's memory allocation recommendation and keep aware of your memory utilization once the device is in production.*** While Cisco provides memory guidelines, they are not natural law. If you have deployed the wrong router with the wrong resources, be assured there will be problems.

There is also one other key dataset stored in the router's RAM that is not a direct component of the IOS. This is the router's system configuration. Once the IOS is loaded up and running, the router needs a number of operating parameters configured before the router can actually perform any useful function. Like the IOS, the router's configuration data is also stored as a single binary file named "startup-config" that is stored and loaded from the router's NVRAM or FLASH if need be. Once the configuration data is loaded into RAM, the configuration is named "running-config." The running-config file is modified whenever a change is made to the router through the CLI or over the network using TFTP, FTP, RCP, and SNMP and, in certain IOS versions, SCP. All changes made to the router

Cool IOS Commands

Michael J. Martin

configuration are applied instantly. However, new changes are only carried down to the startup-config by executing a <write memory> command or a <copy running-config startup-config>.

Controlling the Router Boot Process

How the router determines which IOS and configuration data to load is determined by two factors. The primary attribute is the router's configuration-register setting. The secondary is the <boot> configuration setting contained in the "boot-start" section of the startup-config. These two variables in combination inform the router of the IOS image and configuration data it should when the router boots. Before we get into the details of how to change the boot behavior of the router, let's review the router's "default" bootstrap behavior:

1. System POST (Power-On Self Test)
2. Bootstrap code is loaded from ROM
3. The rommon configuration register (conf-reg) value is read
4. The startup-config is checked for boot system attributes. If they exist they are executed.
5. If no boot system variables exist or they fail, the router falls back first to the local flash/disk storage and loads the first IOS image it discovers (if multiple IOS images exist). If no local IOS image exists, the router will attempt a local broadcast TFTP boot. If the TFTP boot fails, the router will load the ROM based IOS, if the system supports it. Otherwise the router will return to rommon, and an X-modem or TFTP boot can be specified.
6. After IOS has been loaded, the router will load the startup-config into RAM called running-config. If additional configuration commands are defined by boot system attributes, they will be loaded as well. If no startup-config exists on the NVRAM, the router initializes the "initial setup" script.

Now that we have some idea of how the router behaves when it follows the default boot process, let's take a look at the prime mover of the router's boot behavior -- the configuration-register.

Setting The Configuration-Register And The ROM Monitor (Rommon)

The only reason to make changes to the configuration-register value is in the case of two catastrophic events (now you get the prime-mover reference). The first case is when the IOS has become corrupted. The second is when you have forgotten the password on the router. In the case of the former, be prepared for a painful process of loading the IOS over the console port via X-modem (yuck). In the event of the latter, you should be ashamed of yourself (unless the router was hacked) for not sharing information of that importance with at least one other person on your team. At the very least, you should have the information on paper, stored in a safe place.

Even though you should never need to modify it you may want to know what the configuration-register actually is. The configuration-register is a 16-bit value that sets a number of low-level diagnostic and operational values including:

- Boot image location

Cool IOS Commands

Michael J. Martin

- Enable/disable hardware diagnostic mode
- Console baud rate
- System configuration location

In its first incarnation, the configuration-register was a series of jumpers on the AGS system board. As the Cisco product line evolved, it was integrated as an NVRAM value accessible through the ROM monitor or the IOS. To see the router's current configure-register setting, type the IOS exec command

```
<show hardware> or <show version>:
```

```
testrouter-3640#sh hard
```

```
Cisco Internetwork Operating System Software
```

```
IOS (tm) 3600 Software (C3640-IK9O3S-M), Version 12.2(8)T, RELEASE SOFTWARE (fc2)
```

```
TAC Support: http://www.cisco.com/tac
```

```
Copyright (c) 1986-2002 by Cisco Systems, Inc.
```

```
Compiled Wed 13-Feb-02 13:28 by ccai
```

```
Image text-base: 0x60008930, data-base: 0x61714000
```

```
ROM: System Bootstrap, Version 11.1(20)AA2, EARLY DEPLOYMENT RELEASE SOFTWARE (fc1)
```

```
testrouter-3640 uptime is 3 weeks, 1 hour, 7 minutes
```

```
System returned to ROM by power-on
```

```
System image file is "flash:c3640-ik9o3s-mz.122-8.T.bin"
```

```
cisco 3640 (R4700) processor (revision 0x00) with 125952K/5120K bytes of memory.
```

```
Processor board ID 17883986
```

```
R4700 CPU at 100Mhz, Implementation 33, Rev 1.0
```

```
Bridging software.
```

```
X.25 software, Version 3.0.0.
```

```
SuperLAT software (copyright 1990 by Meridian Technology Corp).
```

```
3 FastEthernet/IEEE 802.3 interface(s)
```

```
DRAM configuration is 64 bits wide with parity disabled.
```

```
125K bytes of non-volatile configuration memory.
```

```
32768K bytes of processor board System flash (Read/Write)
```

```
4096K bytes of processor board PCMCIA Slot1 flash (Read/Write)
```

```
Configuration register is 0x2102
```

```
testrouter-3640#
```

Modifying the configuration-register can be accomplished either through rommon prior to the IOS booting or through the global configuration mode when the IOS is running. To modify the configure-register setting through rommon, restart the router and send a terminal break (alt-b) during the IOS loader sequence:

```
System Bootstrap, Version 12.2(8r)T2, RELEASE SOFTWARE (fc1)
```

```
TAC Support: http://www.cisco.com/tac
```

Cool IOS Commands

Michael J. Martin

Copyright (c) 2002 by cisco Systems, Inc.

c3725 processor with 131072 Kbytes of main memory

Main memory is configured to 64 bit mode with parity disabled

Readonly ROMMON initialized

program load complete, entry point: 0x80008000, size: 0xb2a0

program load complete, entry point: 0x80008000, size: 0xb2a0

program load complete, entry point: 0x80008000, size: 0x727c30

Self decompressing the image : #####

monitor: command "boot" aborted due to user interrupt

rommon 1 >

This will drop you into the rommon shell. You then type <confreg>, followed by the configuration-register value (see the different value options bellow) you want the router to boot with:

rommon 9 > confreg 0x2142

You must reset or power cycle for new config to take effect:

rommon 10 >

Once the configure-register value is changed, the router needs to be power-cycled or reset in order to take effect. To reset the router type <reset> at the rommon prompt:

rommon 10 > reset

The router should then behave in accordance to the new configure-register setting (see different setting options below.) To change the configuration-register setting in the IOS, enter global configuration mode:

testrouter-3640#config t

Enter configuration commands, one per line. End with CNTL/Z.

testrouter-3640(config)#

Then type <config-register>, followed by the new configuration-register setting:

testrouter-3640(config)#config-register 0x2142

Then exit out of configuration mode:

testrouter-3640(config)#^Z

When the change is complete, save the running-config to the starting-config and restart the router for the new configuration-register setting to take effect.

Like most IOS functions, there is always more than one option and they differ depending on the hardware platform. The configuration-register is no different. Here are a number of configuration-register settings that can be utilized to set the boot behavior of the router:

- **0x2002** - Sets the router to boot normally, but enables the "break" (alt-b) option to drop into rommon after the router has booted. Use this option when debugging only. Leaving it enabled on a production router is quite risky. Once you have dropped into rommon, you can return to the IOS on some platforms with the command <continue>.

Cool IOS Commands

Michael J. Martin

- **0x2010** - Sets the router check NVRAM for configuration data and boot directly into rommon. To boot the router, type <boot> or <boot {image filename}>. The "break" option is also enabled. This is also a "debugging" option. Once rommon has loaded, you can boot the router by typing the <boot> command. You can see the available IOS images using the <dir {device:}>. To boot a specific image, type <boot> followed by the device and filename.
- **0x8000** - Sets the router to boot in diagnostic mode on 7x00, 4x00, and 2x00 routers. This loads you into rommon. There you can then execute the <priv> command to have full access to the diagnostic tools. *Be very careful here. You can easily damage your router if you don't know what you're doing.*
- **0x2101** - Sets the router to boot off of the IOS ROM (7x00, 25xx and 4x00 routers). Provided your router supports this option. Otherwise the router will boot the first IOS image on the FLASH/ATA file system.
- **0x2102** - The default conf-reg setting, The router processes any "boot system" definition and then follows the default boot process order (FLASH:TFTP:ROM).
- **0x2141** - Sets the router to load the ROM IOS version and ignore the NVRAM startup-config. This setting is used when recovering from a corrupted IOS or a password lockout on an IOS ROM option router.
- **0x2142** - Sets the router to load the IOS from flash and ignore the NVRAM startup-config, booting the router into initial configuration mode. This is the Cisco recommended value for performing password lockout recovery.

Password Recovery: The Quick Version

There are a number of excellent IOS password recovery resources on the Internet by Cisco and others. But no discussion of rommon and the configuration-register would be complete without a quick overview of router password recovery. So, with that said, we'll just cover the basic process and then move on to covering the boot command options.

1. With a console cable connected, restart the router and send a terminal break during the initial IOS load. This will drop you into rommon.
2. At the rommon prompt type: <confreg 0x2142>, followed by <reset>
3. The router will reboot and start "initial configuration mode." When asked, "Would you like to enter the initial configuration dialog?" type "no."
4. This will drop you to the privilege level 1 exec shell. Type "enable" to enter privilege mode 15. At the enable prompt, "#" type "copy startup-config running-config." This loads your old router config, with you at privilege level 15.
5. Once the configuration is loaded, type "configure terminal." At the configuration mode prompt "(config)#", type "enable secret <new password>" or <enable password <new password>" depending on which method you use.

Cool IOS Commands

Michael J. Martin

6. Exit out of configuration mode.
7. Write the new password changes to the startup-config, type "copy running-config startup-config."
8. Now verify the password change. Type "disable." You should now be in the privilege level 1 exec shell. Now type "enable." You will be prompted for the new password. Send it. You should now be back at privilege level 15.
9. The final step is to set the configuration-register setting back to the default. Type "configure terminal" at the configure prompt type "config-register 0x2102." Save the configuration once again. Type "copy running-config startup-config." Once the copy is complete, restart the router.

Working With Internetworking File System (IFS)

The <boot> configuration syntax and commands have changed as the IOS has evolved. This has resulted in the depreciation of some boot options and functional inconsistency with others. This is due largely in part to the adoption of the IFS (introduced in 12.x) and updated file system management (introduced in IOS 11.x) tools. When discussing the <boot> command options, we need to review the new file system management commands and the adoption of the universal resource locator (URL) metaphor. Here is a listing of the URL prefixes utilized by the IFS and file system management tools:

- <copy {src-device:file dst-device:file}> - Copies files to and from different IOS supported file system devices
- <<flash:> - The router's local FLASH file system. Cisco routers utilize three types of flash memory and flash file system types (with the end result being that certain IFS command only work on Type A based hardware):
 1. Type A (ATA Compact Flash) is utilized by the Cisco 7x00, 12x000, 17x0, 37x0 routers and 4x00 and 6x00 switches
 2. Type B (Linear Flash) is utilized by Cisco 16x0, 2x00, 36x0, 4x00 routers and AS5x00 access servers
 3. Type C (ATA DISK) is utilized by the Cisco MC3810 and SC3640
- <disk{0|1}> - ATA Compact Flash Drives used by Type A
- <null:> - Output to a null file system
- <nvr:> - Default storage of the "startup-config"
- <slot {0|1}> - PCMCIA slots on 3x00 and 7x00 series hardware
- <system:> - URL reference for manipulating the startup-config and running-config files.
- <ftp:> - File Transfer Protocol Format: ftp://username:password@host/dir/filename
- <rcp:> - Remote Copy Protocol. Format: rcp://username@host/dir/filename
- <tftp:> - Trivial File Transfer Protocol. Format: tftp://host/dir/filename
- <scp:> - Secure copy. Format: scp://username:password@host/dir/filename

Along with the URL prefixes -- IOS 11.x also introduced an advanced set of file system manipulation commands:

Cool IOS Commands

Michael J. Martin

<dir {fs:} or {/all}> - Prints out the listing of a specific or all available flash file systems (if multiple file system exist) with the "/all" option in place of a specific file system value. Here is an example:

```
CC2217919-C#dir /all
Directory of flash:/
```

```
 1 -rw-   9492852  Dec 8 2003 23:04:30 -05:00 c1700-k9o3sy7-mz.123-5.bin
 2 -rw-     699  Dec 15 2003 21:33:30 -05:00 aaa-config
```

16515072 bytes total (7021392 bytes free)

```
CC2217919-C#
```

!# <pwd> - Prints out the current file system. Here is a syntax example:

```
CC2217919-C#pwd
```

```
flash:/
```

```
CC2217919-C#
```

!# <cd {fs;}> - Used to change between file systems. Here is a syntax example:

```
CC2217919-C#cd flash:
```

```
CC2217919-C#pwd
```

```
flash:/
```

```
CC2217919-C#
```

<delete {fs:file name}> - Deletes a file from the file system. The command behaves differently depending on the file system it is used on. On NVRAM file systems, the file is immediately purged. On FLASH file systems, the file is "marked" as deleted. This allows the file to be recovered with the <undelete {fs:file name}> command. In order to purge the file (and free up actual space on the file system) the <squeeze {fs:file name}> command must be run to purge the file. Here is an example:

```
CC2217919-C#delete nvram:old-CC221-config
```

```
Delete filename [old-CC221-config]?
```

```
Delete nvram:old-CC221-config? [confirm]
```

```
CC2217919-C#
```

<format {fs}> - Erases the file system.

<rename {fs:file name}> - Renames a file.

<more {fs:file name}> - The <more> command functions in quite the same way the Unix version does.

For non-Unix users, <more> is a file parser that allows you to move forward or backwards through any text file. This tool allows you to view any text file on a local or remote file system. Here is an example of looking at another router's configuration backup on the TFTP server:

```
router3640#more tftp://172.30.71.5/outland-rs01-config
```

```
!
```

```
! Last configuration change at 23:32:19 EST Fri Nov 28 2003 by root
```

```
! NVRAM config last updated at 19:58:26 EST Fri Nov 28 2003 by root
```

```
!
```

```
version 12.1
```

```
no service pad
```

Revised December 17, 2003

Page 7 of 9

Cool IOS Commands

Michael J. Martin

```
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
```

Now that you are familiar with the IOS IFS file system management commands, we can have the brief discussion about boot commands.

Setting Up Alternative Boot And Configuration Options

The "boot start" statements are located in the top portion of the router's startup-config following the <hostname> value. As of IOS 12.3, this configuration region is now indicated with the < boot-start-marker> and < boot-end-marker> entries. Here's an example:

```
version 12.3
service timestamps debug datetime msec
service timestamps log datetime localtime show-timezone
service password-encryption
!
hostname CC2217919-C
!
boot-start-marker
boot host tftp:[[172.16.100.1]/CC2217919-config]
boot network tftp start//net-config 172.30.71.5
boot system flash:c1700-k9o3sy7-mz.123-5.bin
boot-end-marker
```

We briefly covered the use of the <boot> command in configuration-register and rommon section above. Now we'll take a look at how the <boot> configuration option is implemented in the IOS. There are four boot option commands:

1. <boot system {tftp|flash|ftp|rcp|rom} {device:filename or filename}> - Sets the IOS image file location from which the router should boot. It is possible to have more than one <boot system> entry on in the <boot-start> config. This provides the flexibility to set one or more common network boot image server that routers can access off the network, making IOS upgrades as simple as changing a file. Just reboot the router and you are upgraded. The local flash: file system can contain a backup image that can be booted in the event of a network failure. (In the event the boot commands cannot be executed, the default boot order is followed.) The <boot system> command can also be used to indicate which IOS image to boot when multiple IOS images exist on the router's flash: file system. When using this command on different hardware platforms, be sure to test first. *Note of caution: The <boot system> command behaves differently between platforms requiring strict or loose syntax.*
2. <boot host {remote URL:filename}> - Sets an alternative remote (to nvram:startup-config) router configuration to load at boot. The command is useful when testing different IOS feature requirements. It allows you load and save changes to alternative configurations without having to modify the default startup-config. The configuration files need to be on a local file system, either flash: or nvram: There is an option to load off of a remote file system, but this options is not

Cool IOS Commands

Michael J. Martin

supported across all hardware platforms and is not available on every IOS feature set. Caution should also be taken when using alternative boot files! Executing a <copy system:running-config system:startup-config> will write the alternative boot configuration to the nvram:startup-config file. So you need to be sure to indicate the correct file name when you save your changes.

Here are examples of the command syntax with the different remote file system types:

```
ftp:[[[//[username[:password]@]location]/directory]/filename]
scp:[[[//[username@]location]/directory]/filename]
tftp:[[[//[location]/directory]/filename]
scp:[[[//[username[:password]@]location]/directory]/filename]
```

3. <boot config local URL:filename> - This command only works on hardware platforms that support Type A Flash File Systems. It allows a local alternative configuration file to be specified. Just as with the <boot host> command, caution should be taken when using alternative boot files! Executing a <copy system:running-config system:startup-config> will write the alternative boot configuration to the nvram:startup-config file. So you need to be sure to indicate the correct file name when you save your changes.
4. <boot network {remote URL:filename}> This command defines network-wide configuration file location. It provides a common place to define global configuration variables that need to be set network-wide. It uses the same configuration syntax format as the <boot host> command:

```
ftp:[[[//[username[:password]@]location]/directory]/filename]
rcp:[[[//[username@]location]/directory]/filename]
tftp:[[[//[location]/directory]/filename]
scp:[[[//[username[:password]@]location]/directory]/filename]
```

Conclusion

Well, that about covers setting router bootstrap options. I hope that you found this article informative and a good use of your time. As always, questions, comments and article topic suggestions are welcome.