Cisco IOS Software Integrity Assurance

Archived: 2026-04-05 14:07:31 UTC

Contents

Introduction

This document analyzes methods that may be used to compromise Cisco devices, including the injection of malicious software in Cisco IOS Software, and describes ways to verify that the software on a Cisco router, both in device storage and in running memory, has not been modified. Additionally, the document presents common best practices that can help protect against attempts to modify hardware or inject malicious software (also referred to as malware) in a Cisco IOS device.

Note: This document applies only to Cisco IOS Software and to no other Cisco operating systems.

In the past, attackers were primarily targeting infrastructure devices to create a denial of service (DoS) situation. While these types of attacks still represent the majority of attacks on network devices, attackers are now looking for ways to subvert the normal behavior of infrastructure devices due to the devices' privileged position within the IT infrastructure.

In fact, by owning an infrastructure device such as a router, the attacker may gain a privileged position and be able to access data flows or crypto materials or perform additional attacks against the rest of the infrastructure.

Malware is software created to modify a device's behavior for the benefit of a malicious third party (attacker). One of the characteristics of effective malware is that it can run on a device stealthily in privileged mode. Malware is usually designed to monitor and exfiltrate information from the operating system on which it is running without being detected. Potentially, sophisticated Cisco IOS malware could attempt to hide its presence by modifying Cisco IOS command output that would normally reveal information about the malware's presence.

An additional property of a malware is the capability to be remotely programmable from Command and Control (C&C) server. Methods to identify possibly compromised infrastructure devices by using telemetry data are discussed in the Telemetry-Based Infrastructure Device Integrity Monitoring white paper.

In general, malware can be installed by using various methods: by exploiting vulnerabilities on the system, or by manipulating an authorized user via social engineering attacks.

On Cisco devices running Cisco IOS Software, a limited number of infection methods are available to malware. Malicious software in Cisco IOS Software may be introduced in the following ways:

- By altering the software image stored on the onboard device file system. This type of malware would be persistent and remain after a reboot.
- By tampering with Cisco IOS memory during run time.
- By modifying the ROM monitor on systems with flash-based ROM monitor storage.
- By exploiting a vulnerability.
- By modifying hardware components of a Cisco IOS device.
- By a combination of some or all of the preceding methods.

Potential Attack Methods

To install malware in Cisco IOS Software, attackers may try to use one of the methods described in this section.

Cisco IOS Software implements several techniques, including the use of safe coding libraries, Address Space Layout Randomization (ASLR), digitally signed software, and Cisco Secure Boot to help protect against memory and code manipulation and provide assurances of authenticity. Administrators should make sure their hardware and software supports these features to ensure protection of the integrity of the device.

However, these technologies will not protect Cisco IOS Software from unauthorized access due to compromised credentials. It is therefore important that administrators protect credentials for privileged accounts (for example, privilege 15) with appropriate controls and by implementing credentials management policies.

Note that an attacker with administrative access to a device, be it a Cisco device or one from any other vendor, can perform activities that may be dangerous or disruptive. Given the state of the current Cisco IOS integrity protection technology, attacks will often exploit inadequacies in secure configuration and network design likely by trying to obtain administrative access.

Commands

Some Cisco IOS devices offer sets of commands that are intended to be used by Cisco Technical Assistance Center (TAC) engineers when troubleshooting a technical problem. Such advanced troubleshooting and diagnostic commands require privileged EXEC level and valid credentials to execute. If these device credentials are compromised, an attacker may be able to use the commands to inject code in memory during run time and modify the behavior of a Cisco IOS device.

It is important to note that not all Cisco IOS platforms offer advanced diagnostic commands. Of the platforms that do, only a very limited set of such commands is usually available. Following common authentication and command authorization security best practices and protecting administrator credentials will help prevent attackers from attempting to install malicious software in Cisco IOS Software. Such best practices are discussed in the Security Best Practices section of this document.

Manipulating Cisco IOS Images

It is possible that an attacker could insert malicious code into a Cisco IOS Software image and load it onto a Cisco device that supports the image. This attack scenario applies to any computing device that loads its operating system from an external, writable device. Even though such a scenario is not impossible, there are image verification techniques, discussed in the Cisco IOS Image File Verification section of this document that could prevent the router from loading such an image.

Additionally, Cisco IOS offers some additional protection for Cisco IOS device and software releases that support digitally signed Cisco IOS Software and Cisco Secure Boot technology. In all cases, such types of attack would require privileged access to the target device.

Vulnerabilities

As with every operating system, it is possible that a vulnerability could exist in Cisco IOS Software that, under certain conditions, could allow malicious code execution. In such a scenario, an attacker who exploited the vulnerability could install or run malicious code in Cisco IOS Software. The Cisco Product Security Incident Response Team (PSIRT) identifies, manages, and discloses all vulnerabilities in and fixes for Cisco products. Any vulnerability that Cisco is made aware of is investigated and disclosed in accordance with the Cisco vulnerability disclosure policy.

The table below summarizes the possible attack methods, the privileges required to perform the attack, and the recommended best practices that, if followed, would greatly reduce the chance of a successful attack.

Possible Attack Methods		
Attack Vector	**Privileges Required**	**Recommended Best Practices**
Code injection in run time via IOS commands	Administrative privileges	Use Authentication, Authorization, and Accounting Use TACACS+ Authorization to Restrict Commands Implement Credentials Management Implement Configuration Controls
Modified Binary Image	Administrative privileges	Use Cisco Secure Boot Deploy Digitally Signed Cisco IOS Images Maintain Cisco IOS Image File Integrity Implement Change Control Harden the Software Distribution

		Server Implement Credentials Management Protect Interactive Access to Devices
Modified ROMMON Image	Administrative privileges	Use Cisco Secure Boot Deploy Digitally Signed Cisco IOS Images Implement Change Control Harden the Software Distribution Server Implement Credentials Management Protect Interactive Access to Devices
Hardware Modification	Physical access to the device	Cisco Value Chain Security
Vulnerabilities that could cause writing in memory	Depends on the vulnerability	Keep Cisco IOS Software Updated

Identification Techniques

This section describes methods that can identify the modification of Cisco IOS image files and run-time memory. The absence of indicators of compromise using these methods may not guarantee that the Cisco IOS device is free from compromise. Readers should note that when facing potential exploitation, the chain of custody becomes important. Administrators need to be aware of chain of custody through all forensic activities, including those presented below, because an exploit could alter specific forensic outputs that would further affect the forensic analysis.

Note: The examples in this document, unless otherwise noted, are taken from a Cisco 7600 Series device using Route Switch Processor 720 and running Cisco IOS 15.1(3)S3 advanced IP services. The output on your device may differ based on device model, operating system release, and feature set. In addition, the commands in this document may use a different syntax, or they may not be present at all. The Cisco recommendation is to follow up with your support organization if you need details about implementing these recommendations for a specific combination of device model, Cisco IOS release, and feature set.

Cisco IOS Image File Verification

Network administrators can use one of several security features to verify the authenticity and integrity of Cisco IOS Software images in use on their network devices. It is also possible to use a process that does not rely on features in Cisco IOS Software.

The following sections contain information about Cisco IOS Software features and administrative processes that can be used to verify the authenticity and integrity of a Cisco IOS Software image.

Using the Message Digest 5 File Validation Feature

The Message Digest 5 (MD5) File Validation feature allows network administrators to calculate the MD5 hash of a Cisco IOS Software image file that is loaded on a device. It also allows administrators to verify the calculated MD5 hash against that provided by the user. After the MD5 hash value of the installed Cisco IOS image is determined, it can also be compared with the MD5 hash provided by Cisco to verify the integrity of the image file.

Note: The MD5 File Validation feature can be used only to check the integrity of a Cisco IOS Software image that is stored on a Cisco IOS device. It cannot be used to check the integrity of an image running in memory.

MD5 hash calculation and verification using the MD5 File Validation feature can be accomplished using the following command:

```
verify /md5 filesystem:filename [md5-hash]
```

Network administrators can use the **verify /md5** privileged EXEC command to verify the integrity of image files that are stored on the Cisco IOS file system of a device. The following example shows how to use the **verify /md5** command on a Cisco IOS device:

```
router# verify /md5 sup-bootdisk:c7600rsp72043-advipservicesk9-mz.151-3.S3
.....<output truncated>.....Done!
verify /md5 (sup-bootdisk:c7600rsp72043-advipservicesk9-mz.151-3.S3) = e383bf779e1373678395
router#
```

Network administrators can also provide an MD5 hash to the **verify** command. If the hash is provided, the **verify** command will compare the calculated and provided MD5 hashes as illustrated in the following example:

```
router# verify /md5 sup-bootdisk:c7600rsp72043-advipservicesk9-mz.151-3.S3 e383bf779e137367
.....<output truncated>.....Done!
Verified (sup-bootdisk:c7600rsp72043-advipservicesk9-mz.151-3.S3) = e383bf779e137367839593e
router#
```

If the network administrator provides an MD5 hash that does not match the hash calculated by the MD5 File Validation feature, an error message will be displayed. This message is shown in the following example:

```
router# verify /md5 sup-bootdisk:c7600rsp72043-advipservicesk9-mz.151-3.S3 0c5be63c4e339707
.....<output truncated>.....Done!
%Error verifying sup-bootdisk:c7600rsp72043-advipservicesk9-mz.151-3.S3
Computed signature  = e383bf779e137367839593efa8f0f725
```

```
Submitted signature = 0c5be63c4e339707efb7881fde7d5324
router#
```

In the preceding examples, the **verify /md5** command calculates and displays the MD5 hash for the entire Cisco IOS image file. This approach is in contrast to the updated **verify** command in the Image Verification feature, which calculates the hash for the entire Cisco IOS image as well as specific portions of the uncompressed Cisco IOS image file.

The **verify** privileged EXEC command was originally introduced for the MD5 File Validation feature and updated by the Image Verification feature to verify the integrity of image files that are stored locally on a device. It can be used to obtain information about the image hashes. The following example demonstrates how to use the updated **verify** command on a Cisco IOS device:

```
router# verify sup-bootdisk:c7600rsp72043-advipservicesk9-mz.151-3.S3
Verifying file integrity of sup-bootdisk:c7600rsp72043-advipservicesk9-mz.151-3.S3
.....<output truncated>.....Done!
Embedded Hash  MD5 : FCEBD3E1AF32221091E920D5960CAE45
Computed Hash  MD5 : FCEBD3E1AF32221091E920D5960CAE45
CCO Hash       MD5 : E383BF779E137367839593EFA8F0F725
Signature Verified
router#
```

In the preceding output, three MD5 hash values are displayed by the **verify** command. The following is an explanation of each MD5 hash value:

- **Embedded Hash:** MD5 hash stored by Cisco in a section of the Cisco IOS image file during the image build process; used to verify section integrity for the Cisco IOS Software image file. This MD5 hash value is calculated for certain sections of the Cisco IOS image file.
- **Computed Hash:** MD5 hash that the Image Verification feature calculates for certain sections of the Cisco IOS Software image file when the **verify** command is executed. This value should be the same as the Embedded Hash to verify section integrity of the Cisco IOS image file. If this value is not equal to the Embedded Hash, the Cisco IOS image file may be corrupted or intentionally altered.
- **CCO Hash:** MD5 hash for the entire Cisco IOS image file. This hash is computed by the **verify** command and is not stored in the Cisco IOS Software image. This value should match the value provided in the Support and Downloads area of the Cisco.com website for this image.

For additional information, see the Image Verification section of Cisco IOS User Security Configuration Guide.

Using the Image Verification Feature

The Image Verification feature builds on the MD5 File Validation functionality to allow network administrators to more easily verify the integrity of an image file that is loaded on the Cisco IOS file system of a device. The purpose of the Image Verification feature is to ensure that corruption of the Cisco IOS Software image file has not

occurred. The corruption detected by this feature could have occurred at any time, such as during the download from Cisco.com or the installation process.

Note: The Image Verification feature does not check the integrity of the image running in memory.

Cisco IOS Software image file verification using this feature can be accomplished using the following commands:

- **file verify auto**
- **copy** [**/erase**] [**/verify** | **/noverify**] *source-url destination-url*
- **reload** [**warm**] [**/verify** | **/noverify**] [*text* | **in** *time* [*text*] | **at** *time* [*text*] | **cancel**

Network administrators can use the **file verify auto** global configuration command to enable verification of all images that are either copied using the **copy** privileged EXEC command or loaded using the **reload** privileged EXEC command. These images are automatically verified for image file integrity.

The following example shows how to configure the **file verify auto** Cisco IOS feature:

```
router# configure terminal
router(config)# file verify auto
router(config)# exit
router#
```

In addition to **file verify auto**, both the **copy** and **reload** commands have a **/verify** argument that enables the Image Verification feature to check the integrity of the Cisco IOS image file. This argument must be used each time an image is copied to or reloaded on a Cisco IOS device if the global configuration command **file verify auto** is not present.

For information about the **copy /verify** and **reload /verify** commands, see the Image Verification section of Cisco IOS User Security Configuration Guide.

Using Offline Image File Hashes

For a file stored on an administrative workstation, a network administrator can verify the MD5 or SHA-512 hash for that Cisco IOS image file using an MD5 or SHA-512 hashing utility. Examples of such utilities are md5sum or sha512sum. Additionally, the size of the Cisco IOS image file can be obtained using the **ls** command on Linux and BSD operating systems and the **dir** command on Microsoft Windows platforms.

The following example demonstrates the MD5 calculation and file size display for Linux-based systems:

```
$
$ md5sum 7600rsp72043-advipservicesk9-mz.151-3.S3.bin e383bf779e137367839593efa8f0f725
  7600rsp72043-advipservicesk9-mz.151-3.S3.bin
$
$ ls -l
 7600rsp72043-advipservicesk9-mz.151-3.S3.bin -r--r--r--   1 user user 167167524 May 16 15:
```

```
  7600rsp72043-advipservicesk9-mz.151-3.S3.bin
$
```

The following example shows the use of the **fsum** utility on a Windows system:

```
C:\>fsum -md5 7600rsp72043-advipservicesk9-mz.151-3.S3.bin

SlavaSoft Optimizing  Checksum Utility - fsum 2.52.00337
Implemented using SlavaSoft  QuickHash Library lt;www.slavasoft.com>
Copyright (C) SlavaSoft Inc.  1999-2007. All rights reserved.

; SlavaSoft Optimizing  Checksum Utility - fsum 2.52.00337 <www.slavasoft.com>
;
; Generated on 05/20/08 at  00:01:13
;
e383bf779e137367839593efa8f0f725 *7600rsp72043-advipservicesk9-mz.151-3.S3.bin
```

Note: The use of the **fsum** utility is for illustrative purposes only and should not be interpreted as an endorsement of the tool.

Using Information from Cisco.com

When the hash and file size for a Cisco IOS Software image have been collected, network administrators can verify authenticity of the image using information provided in the Support and Downloads area on the Cisco.com website. This provides details about each publicly available IOS image and may require a valid Cisco.com account.

Network administrators must identify their Cisco IOS Software release (this can be done by using information obtained from output provided by the **show version** command) and navigate through the Downloads area to locate the image in use on the Cisco IOS device. Network administrators should verify that one of the following hashes matches the MD5 hash that is provided on Cisco.com:

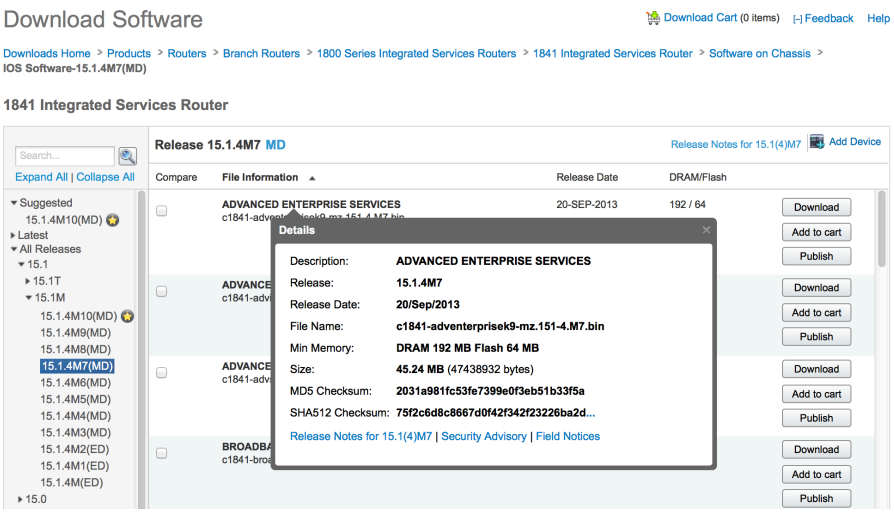
- CCO hash calculated by the Cisco IOS **verify** command (part of the Cisco IOS Image Verification feature)
- MD5 hash calculated by the **verify /md5** command (part of the MD5 File Validation Cisco IOS feature)
- MD5 or SHA-512 hash calculated by a third-party utility

If the MD5 or SHA-512 hash value for the whole Cisco IOS image file does not match the value provided by Cisco, network administrators should download the Cisco IOS image file from the Cisco IOS Upgrade Planner and use the file verification methods described in this document to verify integrity of the Cisco IOS image file.

As of June 1, 2015, Cisco is providing the SHA-512 hash value for all Cisco IOS Software releases that are published on Cisco.com. Administrators can compare the SHA-512 value published on Cisco.com and the SHA-512 value calculated by a third-party utility to verify the integrity of a Cisco IOS image.

The following is an example of the information provided on Cisco.com during one of the steps required for downloading Cisco IOS Software Release 15.14M7 for a Cisco 1841 Integrated Services Router:

Figure 1. Download Software



Verifying Authenticity for Digitally Signed Images

Cisco IOS supports digitally signed images on some platforms. Digitally signed Cisco software is digitally signed using secure asymmetric (public-key) cryptography.

Digitally signed Cisco software increases the security posture of Cisco IOS devices by ensuring that the software running in the system has not been altered and originates from a trusted source.

Administrators can verify the authenticity and integrity of the binary file by using the **show software authenticity file** command. In the following example, taken from a Cisco 1900 Series Router, the command is used to verify the authenticity of *c1900-universalk9-mz.SPA.152-4.M2.bin* on the system:

```
Router# show software authenticity file c1900-universalk9-mz.SPA.152-4.M2

File Name                   : c1900-universalk9-mz.SPA.152-4.M2
Image type                  : Production
    Signer Information
        Common Name         : CiscoSystems
        Organization Unit   : C1900
        Organization Name   : CiscoSystems
    Certificate Serial Number : 509AC949
    Hash Algorithm          : SHA512
    Signature Algorithm     : 2048-bit RSA
    Key Version             : A
```

In addition, administrators can use the **show software authenticity running** command to verify the authenticity of the image that is currently booted and in use on the device. Administrators should verify that the *Certificate Serial Number* value matches the value obtained by using the **show software authenticity file** on the binary file.

The following example shows the output of **show software authenticity running** on a Cisco 1900 Series Router running the *c1900-universalk9-mz.SPA.152-4.M2* image.

```
Router# show software authenticity running

SYSTEM IMAGE
------------
Image type                    : Production
    Signer Information
        Common Name           : CiscoSystems
        Organization Unit     : C1900
        Organization Name     : CiscoSystems
    Certificate Serial Number : 509AC949
    Hash Algorithm            : SHA512
    Signature Algorithm       : 2048-bit RSA
    Key Version               : A
    Verifier Information
        Verifier Name         : ROMMON 1
        Verifier Version      : System Bootstrap, Version 15.0(1r)M9, RELEASE SOFTWARE (fc1
Technical Support: http://www.cisco.com/techsupport
```

This example also shows that the Certificate Serial Number value, *509AC949*, matches the one obtained with the previous example.

Additional information is in Digitally Signed Cisco Software.

Cisco IOS Run-Time Memory Integrity Verification

Network administrators can also verify the integrity of the run-time memory of Cisco IOS. This is not a trivial task and there is not currently a solution that would allow the network administrator to analyze all parts of memory manually. However, the best way to verify the integrity of run-time memory for Cisco IOS Software is to analyze the region of memory called “main:text.”

The main:text section contains the actual executable code for Cisco IOS Software after it is loaded in memory. As such, verifying its integrity is particularly relevant for detecting in-memory tampering. This region of memory should not change during normal Cisco IOS Software operation, and should be the same across reloads.

Because this region of memory holds the actual operating system code, it should not change between devices as long as they are the same model and running the same release number and feature set. However, if the Cisco IOS release in use is ASLR enabled, these assumptions become invalid. A side effect of ASLR is changing some parts of the operating system code. This means the memory contents will be different across devices, even if they are running the same operating system release and feature set. See the Cisco IOS Address Space Layout Randomization Considerations section of this document for additional information.

Note: The absence of indicators of compromise using the methods presented in this section may not guarantee that the Cisco IOS device was not compromised.

Core Dump

Cisco IOS devices support exporting the contents of the running memory. After the export, comparisons between the running memory dump, also called core dump, and the associated sections in the Cisco IOS image file can be performed to detect modification of the run-time memory contents.

The first step is to create a dump of the run-time memory. Most Cisco IOS releases support a memory dump via the **write core** command. Further information about core dumps on Cisco IOS devices can be found in the Creating Core Dumps support document.

The core dump can be written to an external server via several protocols, including FTP and Remote Copy Protocol (RCP). The following example shows how to configure the Cisco IOS device to write the core dump to an external server via FTP:

```
exception core-file <name> compress
exception protocol ftp
exception region-size 65536
exception dump <ip address of the  ftp server>
ip ftp username <user>
ip ftp password <pass>
```

The core dump process will usually write several files. The file that contains the text region is the one in the format *<core_filename> _<timestamp>*.

Note: The name of the core dump file generated by Cisco IOS Software may differ depending on the specific Cisco IOS device, Cisco IOS release, and feature set in use.

To determine the boundary of the text region in the core dump file, use the **show region** command. The following example is from a Cisco 7600 Series device using Supervisor Engine 720 and running Cisco IOS 12.2(33)SRD4 advanced IP services. It shows that the main:text area starts with an offset of 0x1012B8 (this can be calculated by subtracting the start of main from the start of main:text, that is, 0x401012B8 - 0x40000000) and has a size of 67464520 bytes:

```
router# show region

Region Manager:

     Start          End     Size(b)   Class  Media  Name
0x08000000   0x0BFFFFFF    67108864  Iomem   R/W    iomem
0x40000000   0x4BFFFFFF   201326592  Local   R/W    main
0x401012B8   0x44157FFF    67464520  IText   R/O    main:text
0x4415A1D0   0x4475D44F     6304384  IData   R/W    main:data
```

```
0x4475D450   0x463B8BAF    29734752  IBss    R/W    main:bss
0x463B8BB0   0x4BFFFFFF    96760912  Local   R/W    main:heap
0x50000000   0x5FFF7FFF   268402688  Local   R/W    more_heap
0x80000000   0x8BFFFFFF   201326592  Local   R/W    main:(main_k0)
0xA0000000   0xABFFFFFF   201326592  Local   R/W    main:(main_k1)
```

After the boundaries are determined, extract the text region from the core dump.

There are several tools available for extracting the text region. The Linux utility **dd** is used in this example. To avoid a block error, set the block size (bs) to 1. Additionally, because the text region starts with an offset of 0x1012B8, provide this information to **dd** after converting the offset in decimal equivalent: 1053368. **dd** also needs the size of the region: 67464520.

Note that if the *compress* option is used, the file needs to be uncompressed before using it with **dd**.

```
dd if=<corefile> bs=1 count=67464520 skip=1053368  of=router_main_text
```

The file *router_main_text* will include the text region.

After the text region is isolated, compute the checksum of the file. In this example, the Linux utility **md5sum** calculates the MD5 checksum of the file:

```
md5sum router_main_text
1edd0985da7f1a490729fd0aaf9c0bd7 router_main_text
```

This value should be compared to the MD5 hash value obtained by a hashing a main:text section taken from a router that it is known not to be compromised, also referred as known-good text region. The following section proposes a procedure to create a known-good text region.

This method implies trust in the memory-dumping process, which may be compromised.

Creating a Known-Good Text Region

A known-good text region is a file that contains the main:text that it is known not to be compromised and that can be used as a reference point during the forensic operation. This section proposes the procedure that can be used to create a known-good text region.

1. Download the Cisco IOS image from the Cisco Support and Downloads website and note the MD5 value of the binary.
2. Use the method described in the Using Offline Image File Hashes section of this document to verify that the MD5 hash of the image downloaded matches the one on the Cisco Support and Downloads website. We will call this the "known-good image."
3. Load the known-good image on a Cisco IOS device.
4. Reload the device.

5. After reload, use the method described in the Using the Message Digest 5 File Validation Feature section of this document to verify the integrity of the image that has been booted. We will call this "known-good device." The main:text memory region of this device will be called the known-good text region.
6. Use the method described in Verify MD5 Validation Feature for the Text Region to calculate the MD5 hash value of the main:text region. This is what we call the "MD5 hash of a known-good text region."

One alternative to Steps 5 and 6 would be to generate a core dump from the known-good device and then extract the known-good text region and calculate the MD5 hash using the method described in the Core Dump section of this document.

Text Memory Section Export

An alternative to collecting the core dump is exporting the text section using the **copy** command.

Caution: Due to a bug in Cisco IOS Software, this method may cause a crash and a reload on the following platforms. Therefore this method should not be used for these platforms:

- Cisco Catalyst 6880-X Switch
- Cisco 3900E Series Integrated Services Routers
- Cisco 1000 Series Connected Grid Routers

Depending on the Cisco IOS release, the **copy** command can copy files stored in the Cisco IOS file system to an external server via several protocols, including FTP and Secure Copy Protocol (SCP). The following example shows how to copy the text memory section via FTP.

Configure the FTP username and password if it has not been done already:

```
ip ftp username <user>
ip ftp password <pass>
```

Use the **dir** command to locate the text region. This is usually in the system:/memory directory. The following example shows the output of **dir system:/memory** taken from a Cisco 7600 Series device using Supervisor Engine 720 and running Cisco IOS 12.2(33)SRD4 advanced IP services:

```
router# dir system:/memory

Directory of system:/memory/

  8 -r--   29734752     <no date> bss
  7 -r--    6304384     <no date> data
  9 -r--   96752816     <no date> heap
  4 -r--   67108864     <no date> iomem
  5 -r--  201326592     <no date> main
 11 -r--  201326592     <no date> main_k0
 12 -r--  201326592     <no date> main_k1
```

```
 10 -r--  268402688     <no date> more_heap
  6 -r--   67464520     <no date> text
```

Export the text region by using the **copy** command:

```
router# copy system:memory/text ftp:
Address or name of remote host []? <FTP server ip address>
Destination filename [text]? router_main_text
Writing router_main_text !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
<output suppressed>
```

After the file has been exported, use the procedure described in the Core Dump section to verify the integrity of main text area.

This method implies trust in the copy process, which may itself be compromised.

Verify MD5 Validation Feature for the Text Region

Network administrators can use the **verify /md5** command to compute the MD5 checksum of the text region without creating a memory dump. The text region is usually located in the system:/memory directory.

Caution: Due to a bug in the Cisco IOS Software code, this method may cause a crash and a reload on the following platforms. Therefore this method should not be used for these platforms:

- Cisco Catalyst 6880-X Switch
- Cisco 3900E Series Integrated Services Routers
- Cisco 1000 Series Connected Grid Routers

Use the **dir** command to locate the text region. This is in system:/memory. The following example shows the output of **dir system:/memory** taken from a Cisco 7600 Series device using Supervisor Engine 720 and running Cisco IOS 12.2(33)SRD4 advanced IP services:

```
router# dir system:/memory

Directory of system:/memory/

  8  -r--   29734752    <no date> bss
 7  -r--    6304384     <no date> data
 9  -r--   96752816     <no date> heap
 4  -r--   67108864     <no date> iomem
 5  -r--  201326592     <no date> main
 11 -r--  201326592     <no date> main_k0
 12 -r--  201326592     <no date> main_k1
 10 -r--  268402688     <no date> more_heap
 6  -r--   67464520     <no date> text
```

Use the **verify /md5** command to calculate the MD5 checksum of the text region:

```
router# verify /md5 system:memory/text
 ....................................................................
 [...]
 .....................................Done!
 verify /md5 (system:memory/text) = 1edd0985da7f1a490729fd0aaf9c0bd7
```

This value should be compared to the MD5 hash value obtained by hashing a main:text section taken from a router that it is known not to be compromised, also referred as the known-good text region. See the Creating a Known-Good Text Region section of this document.

This method implies trust in the onboard **verify /md5** command, which may itself be compromised.

Cisco IOS Address Space Layout Randomization Considerations

To help harden the security posture of Cisco IOS, some products run a Cisco IOS image with Address Space Layout Randomization (ASLR).

When ASLR is active, the procedures described above may not be valid because ASLR dynamically changes the memory space where the text and/or other memory regions are loaded at boot time. In some situations, this also means that the instructions in the text region are changed after each reload so the text region does not stay the same across reloads or across different devices.

To determine whether ASLR is active, network administrators can compare the output of the **show region** command between two Cisco IOS devices running the same image and feature set. If the regions have the same starting and ending addresses, that image does not use ASLR on that platform. If the addresses are different, ASLR is active.

The following example shows the output of **show region** for two identical Cisco IOS routers (c1841) running identical images (C1841-ADVENTERPRISEK9-M), Version 15.3(2)T

```
router# show region

Region Manager:

       Start         End      Size(b)   Class  Media  Name
 0x16000000   0x17FFFFFF     33554432   Iomem   R/W   iomem:(iomem)
 0x60000000   0x75FFFFFF    369098752   Local   R/W   main
 0x6001B5B8   0x6487FFFF     75909704   IText   R/O   main:text
 0x6488BC40   0x6692125F     34166304   IData   R/W   main:data
 0x66921260   0x6742621F     11554752   IBss    R/W   main:bss
 0x67426220   0x75FFFFFF    247307744   Local   R/W   main:heap
 0x80000000   0x95FFFFFF    369098752   Local   R/W   main:(main_k0)
 0xA0000000   0xB5FFFFFF    369098752   Local   R/W   main:(main_k1)
 0xF6000000   0xF7FFFFFF     33554432   Iomem   R/W   iomem
```

```
  router# show region

Region Manager:

      Start           End     Size(b)   Class  Media  Name
 0x16000000    0x17FFFFFF    33554432   Iomem   R/W   iomem:(iomem)
 0x60000000    0x75FFFFFF   369098752   Local   R/W   main
 0x6001EDF8    0x6487FFFF    75895304   IText   R/O   main:text
 0x6488F480    0x66924A9F    34166304   IData   R/W   main:data
 0x66924AA0    0x67429A5F    11554752   IBss    R/W   main:bss
 0x67429A60    0x75FFFFFF   247293344   Local   R/W   main:heap
 0x80000000    0x95FFFFFF   369098752   Local   R/W   main:(main_k0)
 0xA0000000    0xB5FFFFFF   369098752   Local   R/W   main:(main_k1)
 0xF6000000    0xF7FFFFFF    33554432   Iomem   R/W   iomem
```

In the preceding example, the starting address of the text, data, and region is different. This information indicates that ASLR is active for this combination of software and hardware.

Additional Indicators of Compromise

In addition to verifying the integrity of the run-time memory, network administrators can check external logs and logs stored on the Cisco IOS device itself for the presence of “unusual” commands.

Unusual and Suspicious Commands

The presence of the following commands should trigger further investigation. The asterisk symbol * indicates any text that follows the command itself.

- **gdb ***
- **test ***
- **tclsh ***
- **debug ***
- **service internal**
- **config-register***
- **boot***
- **upgrade***
- **attach ***
- **remote ***
- **ipc-con ***
- **if-con ***
- **execute-on ***
- **service-monitor ***
- **show region**
- **show memory ***
- **show platform ***

- **do-exec** version of any of the above

Note: Cisco IOS allows command abbreviation. For example, typing **se in** instead of **service internal** will still configure service internal on a device. When checking the logs, abbreviation of commands such as **tes**, **rem**, and **se in** should also be considered.

Checking That Cisco IOS Software Call Stacks Are Within the Text Section Boundaries

During normal operation, Cisco IOS processes should have the program counter (PC) and return address (RA) within the boundary of the text section. If this is not the case, the events should be further investigated.

To verify that the PC and RA are within the text section boundaries, use the **show stack** *pid* command where the process ID (PID) can be obtained, for example, by using the **show process** command. The following example shows how to display the PID of the process running on the Cisco IOS device:

```
Router# show process

 CPU utilization for five seconds: 0%/0%; one minute: 1%; five minutes: 0%
 PID   QTy       PC  Runtime (ms)     Invoked  uSecs    Stacks TTY Process
   1   Cwe  9D38588             0          25      0 5436/6000   0 Chunk Manager
   2   Csp  B698AA4            92       15670      5 2240/3000   0 Load Meter
   4   Mwe  A7F5568             4         268     14 5756/6000   0 Retransmission o
   5   Mwe  A7F30D8             0           4      0 5008/6000   0 IPC ISSU Dispatc
   6   Mwe  B38064C             0           1      0 5728/6000   0 PF Redun ICC Req
   7   Lst  9D63BC0        127108       13231   9606 5436/6000   0 Check heaps
   8   Cwe  9D55360             8        1310      6 5428/6000   0 Pool Manager
   9   Mwe  9D55250             0           1      0 5752/6000   0 DiscardQ Backgro
   [...]
```

Administrators may use the **show stack** command to display information about the PC or RA for each process displayed with the **show process** command. (Depending on the software version and model, the output may include information about a PC or RA.)

The following example shows use of the **show stack** command to reveal RA information for PID 5. The result identifies the IPC ISSU Dispatch process:

```
Router# show stack 5

Process 5: IPC ISSU Dispatch Process
 Stack segment 0x1551EDAC - 0x1552051C
 FP: 0x155204D0, RA: 0x9D6CD44
 FP: 0x15520508, RA: 0xA7F30DC
 FP: 0x15520510, RA: 0xB6A36E4
 FP: 0x0, RA: 0xB69D918
```

After the information about the PC or RA is available, administrators are advised to verify that the memory addresses fall into the text region boundaries. These boundaries can be displayed by using the **show region** command:

```
Router# show region

Region Manager:
      Start        End     Size(b) Class  Media  Name
 0x04000000 0x77FFFFFF  1946157056 Local  R/W    main
 0x04000000 0x07FFFFFF    67108864 Local  R/W    main:rommon
 0x08000000 0x0FFFFFFF   134217728 IText  R/O    main:text
 0x10000000 0x11A7B06F    27766896 IData  R/W    main:data
 0x11A7B070 0x13CA9A1B    35842476 IBss   R/W    main:bss
 0x13CA9A1C 0x77FFFFFF  1681221092 Local  R/W    main:heap
 0x78000000 0x7FFFFFFF   134217728 Iomem  R/W    iomem
 Free Region Manager:
      Start       End      Size(b) Class  Media  Name
```

In this case, all PC addresses 0x9D6CD44, 0xA7F30DC, and 0xB6A36E4 are within the text section boundaries (between 0x08000000 and 0x0FFFFFFF).

Checking Command History in the Cisco IOS Core Dump

Cisco IOS uses an internal buffer to record all commands typed either via the system console or via vty interfaces. When a core dump is generated, the command history buffer is also copied in the core dump file. Network administrators can search the core dump to look for unusual commands.

The following example shows how to search a core dump file by using the Linux utility **string**:

```
$ strings <CORE> |grep ^CMD:
 CMD: 'verify /md5 system:memory/text' 06:59:50 UTC Wed Jan 15 2014
 CMD: 'service internal | i exce' 07:02:41 UTC Wed Jan 15 2014
 CMD: 'conf t' 07:02:45 UTC Wed Jan 15 2014
 CMD: 'exception flash procmem bootflash:' 07:02:54 UTC Wed Jan 15 2014
 CMD: 'exception core-file CORE compress ' 07:03:31 UTC Wed Jan 15 2014
```

The command history can be used to check whether some of the suspicious commands, such as those listed in the Unusual and Suspicious Commands section, have been run on a router, which would be an indication of compromise.

The presence of unusual commands or repetition of commands, even those not listed in the Unusual and Suspicious Commands section, should also be investigated because these commands could indicate a way for an attacker to cover traces.

Checking the Command History

Network administrators can use the **show history all** command to access command history records. The following example shows how to search for the presence of the **service internal** command in the history buffer:

```
router# show history all | include se

 CMD: 'show run | include ^service internal' 09:55:17 UTC Thu Jan 16 2014
 CMD: 'show run | include ^service internal' 10:06:54 UTC Thu Jan 16 2014
 CMD: 'show run | include ^service internal' 10:49:54 UTC Tue Jan 21 2014
 CMD: 'sh run | i service' 17:20:34 UTC Thu Jan 23 2014
 CMD: 'service internal' 10:40:14 UTC Fri Jan 24 2014
 CMD: 'sho history all | i ser' 10:41:00 UTC Fri Jan 24 2014
 CMD: 'show history all | i ser' 10:41:30 UTC Fri Jan 24 2014
 CMD: 'ser in' 10:41:39 UTC Fri Jan 24 2014
 CMD: 'show history all | i ser' 10:41:42 UTC Fri Jan 24 2014
```

The command history can be used to check whether some of the suspicious commands, such as those listed in the Unusual and Suspicious Commands section, have been run on a router, which would be an indication of compromise.

The presence of unusual commands or repetition of commands, even those not listed in the Unusual and Suspicious Commands section, should also be investigated because these commands could indicate a way for an attacker to cover traces.

Checking External Accounting Logs

Cisco IOS can be configured to send accounting information for exec and configuration commands to an external server via the TACACS+ protocol as explained in the Security Best Practices section. Commands can be used to check whether some of the suspicious commands have been run on a router, which would be an indication of compromise.

Checking External Syslog Logs

Cisco IOS Software can be configured to send syslog logs to an external syslog server. Currently Cisco IOS Software will not send commands executed via the console or vty to the syslog server. Network administrators should check the logs for unusual connections or connection attempts to the Cisco IOS devices via vty, console, or other available methods.

Checking Booting Information

Information about the last time the Cisco IOS device was reloaded and the reason may provide additional insight about a possible compromise. For example, an unscheduled reload should raise attention and be investigated further.

Network administrators can use the **show version** command to see information about uptime, last reload cause, and which file was used to boot the Cisco IOS. The following is an example of **show version** output taken from a

Cisco 7606.

Important information in order of appearance in the output is:

- **Router uptime:** This information indicates how long the router has been up. This information can be correlated with change management logs to see whether a reload was authorized and expected.
- **Image booted:** This field provides information about which file was used to boot the Cisco IOS device. Administrators are advised to ensure that the filename matches the Cisco IOS image they intended to boot.
- **Configuration register:** This value is used to indicate how the router should boot. The default value is 0x2102 and should not be modified under normal circumstances. Modification of this value may indicate an attempt to change the correct boot sequence. Additional information is in Use of the Configuration Register on All Cisco Routers.

```
Router# show version

 Cisco IOS Software, c7600rsp72043_rp Software (c7600rsp72043_rp-ADVIPSERVICESK9-M),
 Version 12.2(33)SRD4, RELEASE SOFTWARE (fc2)
 Technical Support: http://www.cisco.com/techsupport
 Copyright (c) 1986-2010 by Cisco Systems, Inc.
 Compiled Mon 22-Feb-10 00:21 by prod_rel_team

   ROM: System Bootstrap, Version 12.2(33r)SRE, RELEASE SOFTWARE (fc1)
BOOTLDR: Cisco IOS Software, c7600rsp72043_rp Software (c7600rsp72043_rp-ADVIPSERVICESK9-M
 Version 12.2(33)SRD4, RELEASE SOFTWARE (fc2)

   Router uptime is 1 day, 21 hours, 41 minutes
 Uptime for this control processor is 1 day, 21 hours, 41 minutes
 System returned to ROM by power-on (SP by power on)
 System image file is "sup-bootdisk:c7600rsp72043-advipservicesk9-mz.122-33.SRD4"
 Last reload type: Normal Reload

     This product contains cryptographic features and is subject to United
 States and local country laws governing import, export, transfer and
 use. Delivery of Cisco cryptographic products does not imply
 third-party authority to import, export, distribute or use encryption.
 Importers, exporters, distributors and users are responsible for
 compliance with U.S. and local country laws. By using this product you
 agree to comply with applicable laws and regulations. If you are unable
 to comply with U.S. and local laws, return this product immediately.

   A summary of U.S. laws governing Cisco cryptographic products may be found at:
 http://www.cisco.com/wwl/export/crypto/tool/stqrg.html

   If you require further assistance please contact us by sending email to
 export@cisco.com.

   Cisco CISCO7606 (M8500) processor (revision 1.0) with 917504K/65536K bytes of memory.
 Processor board ID FOX090407DR
```

```
BASEBOARD: RSP720
CPU: MPC8548_E, Version: 2.0, (0x80390020)
CORE: E500, Version: 2.0, (0x80210020)
CPU:1200MHz, CCB:400MHz, DDR:200MHz,
L1: D-cache 32 kB enabled
I-cache 32 kB enabled

  Last reset from power-on
1 Virtual Ethernet interface
4 Gigabit Ethernet interfaces
4 Ten Gigabit Ethernet interfaces
3964K bytes of non-volatile configuration memory.

  500472K bytes of Internal ATA PCMCIA card (Sector size 512 bytes).
Configuration register is 0x2102
```

Checking the ROM Monitor Variable

The ROM monitor is a bootstrap program that initializes the hardware and boots the Cisco IOS Software.

Because the ROM monitor settings are persistent, information about the ROM monitor variable values could indicate an attempt to influence the Cisco IOS boot sequence. Administrators can use the **set** command while in the ROM monitor prompt to see the value of the ROM monitor variables.

Note: Entering the ROM monitor prompt will require a reload of the Cisco IOS device.

The output of the **set** command may differ depending on the platform and Cisco IOS release; however, administrators should ensure that the following conditions are met:

- The BOOT variable is set, reflecting the image file that should be used to boot Cisco IOS Software
- The OFFSET variable is not set (that is, it does not appear in the output)

The following example shows the output of the **set** command executed on a Cisco 7600 Series device using Supervisor Engine 720 and running Cisco IOS 12.2(33)SRD4 advanced IP services:

```
rommon 1 > set
PS1=rommon ! >
ALLOWANYFAN=1
ALLOWANYPS=1
LOG_PREFIX_VERSION=1
SLOTCACHE=cards;
ADJ_MCAST=
BOOT=disk0:c7600s72033-advipservicesk9-mz.122-33.SRD4,12;
RANDOM_NUM=2050115557
RELOAD_TYPE=1
TYFIB_BLOCK_ALLOC=
NT_K=0:0:0:0
```

```
BSI=0
ACL_DENY=0
PF_REDUN_CRASH_COUNT=0
RET_2_RTS=10:59:51 UTC Wed Feb 5 2014
RET_2_RCALTS=1391597993
CRASHINFO=bootflash:crashinfo_20140205-105949-UTC
?=0
```

Checking the ROM Monitor Information

Administrators can use the **show rom-monitor** command to verify the current version of the ROM monitor and whether the ROM monitor has been upgraded on the Cisco IOS device. The following example shows the output of the command from a Cisco IOS 1800 Series router running Cisco IOS Software release where a ROM monitor upgrade has been performed

```
Router# show rom-monitor
ReadOnly ROMMON version:
System Bootstrap, Version 12.3(8r)T8, RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 2004 by cisco Systems, Inc.
Upgrade ROMMON version:
System Bootstrap, Version 12.4(13r)T, RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 2006 by cisco Systems, Inc.
Currently running ROMMON from Upgrade region
ROMMON from Upgrade region is selected for next boot
```

Note: The output of this command may differ depending on the Cisco IOS hardware platform or software release.

Administrators should make sure that ROM monitor upgrade was a scheduled and legitimate action.

Security Best Practices

Maintain Cisco IOS Image File Integrity

To minimize the risk associated with malicious code, it is important that network administrators develop and consistently apply a secure methodology for Cisco IOS Software image management. This secure process must be used from the time a Cisco IOS Software image is downloaded from Cisco.com until a Cisco IOS device begins using it.

Although processes may vary based on the network and its security and change management requirements, the following procedure represents an example of best practices that may help minimize the possibility of malicious code installation.

- When downloading a Cisco IOS Software image from www.cisco.com, record the MD5 hash as presented on the Support and Downloads page of www.cisco.com.
- After the image has been downloaded to an administrative workstation, the MD5 hash of the local file should be verified against the hash presented by the Cisco IOS Upgrade Planner.
- After the Cisco IOS Software image file has been verified as authentic and unaltered, copy it to write-once media or media that can be rendered as read-only after the image has been written.
- Verify the MD5 hash of the file written to the read-only media to detect corruption during the copy process.
- Remove the local file on the administrative workstation.
- Relocate the read-only media to the file server that is used for Cisco IOS Software image distribution to Cisco IOS devices.
- Transfer the Cisco IOS Software image from the file server to the Cisco IOS device using a secure protocol that provides both authentication and encryption.
- Verify the MD5 hash of the Cisco IOS Software image on the Cisco IOS device using any of the procedures detailed in the Cisco IOS Image File Verification section of this document.
- Modify the configuration of the Cisco IOS device to load the new Cisco IOS Software image upon startup.
- Reload the Cisco IOS device to place the new software into service.

Implement Change Control

Change control is a mechanism through which network device changes are requested, approved, implemented, and audited. Change control is a great help in determining which changes have been authorized and which are unauthorized. Change control is important to help ensure that only authorized and unaltered Cisco IOS Software is used on Cisco IOS devices in the network.

Harden the Software Distribution Server

The server that is used to distribute software to Cisco IOS devices in the network is a critical component of network security. Several best practices should be implemented to help ensure the authenticity and integrity of software that is distributed from this server. These best practices include the following:

- Application of well-established operating system hardening procedures that are specific to the operating system in use
- Configuration of all appropriate logging and auditing capabilities, including logging to write-once media
- Placement of the software distribution server on a secure network with restricted connectivity from all but the most trusted networks
- The use of restrictive security controls to limit interactive access (as an example, SSH) to only a subset of trusted network administrators

Keep Cisco IOS Software Updated

Cisco IOS Software used in the network must be kept up to date so that new security functionality can be used and exposure to known vulnerabilities disclosed through Cisco Security Advisories is minimal.

Cisco is continually evolving the security of Cisco IOS Software images through the implementation of new security functionality and the resolution of bugs. For these reasons, it is imperative that network administrators maintain their networks in a manner that includes using up-to-date software. Failure to do so could expose vulnerabilities that may be used to gain unauthorized access to a Cisco IOS device.

Cisco transparently communicates vulnerabilities found in all Cisco products according to the Cisco Security Vulnerability Policy.

Network administrators can use the Cisco IOS Software Checker tool to search for Cisco Security Advisories that address specific Cisco IOS Software releases.

Deploy Digitally Signed Cisco IOS Images

Digitally signed Cisco software is digitally signed using secure asymmetrical (public-key) cryptography.

The purpose of digitally signed Cisco software is to increase the security posture of Cisco IOS devices by ensuring that the software running in the system has not been tampered with and originated from a trusted source as claimed.

For additional information, see Digitally Signed Cisco Software.

Cisco Secure Boot

Cisco Secure Boot is a secure startup process that your Cisco device performs each time it boots up. Beginning with the initial power-on, a special purpose hardware device, known as the Trust Anchor module, verifies the integrity of the ROMMON code and the IOS image via digital signatures as they each are loaded. If any failures are detected, the user is notified of the error and the device will wait for the operator to correct the error. This prevents your network device from executing tainted network software.

Cisco Value Chain Security

Cisco's Value Chain Security program focuses on counterfeit products, tainted products, and misuse of intellectual property. Just as important as physical security, maintaining a chain of custody from manufacturing through installation and provisioning is vital to a trustworthy network. There are many avenues to introduce malware or fraudulent hardware into network devices, and Cisco's Value Chain Security program ensures that devices delivered with the Cisco Systems name are authentic and unmodified.

While this program will help ensuring the authenticity of the Cisco hardware, administrator should make sure they have tight control on the delivering of new or refurbished equipment once it has arrived on the premises.

For additional information, see Cisco Value Chain Security.

Leverage the Latest Cisco IOS Security Protection Features

Cisco is continuously working on increasing the security protection present in Cisco devices. Whenever possible, Cisco leverages the current hardware and provides software updates that include the latest security protection

features. However, in some cases new hardware capabilities are needed to provide the best protection.

Administrators should review their hardware and software to make sure that features such as Cisco Digitally Signed Image, ASLR, and Cisco Secure Boot are present in devices running in critical segments of their network infrastructure.

Use Authentication, Authorization, and Accounting

The comprehensive implementation of Authentication, Authorization, and Accounting (AAA) is critical to ensuring the security of interactive access to network devices. Furthermore, AAA (specifically the authorization and accounting functions) should be used to limit the actions that authenticated users can perform and provide an audit trail of individual user actions.

The following example shows the necessary configuration to send accounting information to an external AAA server. Note that Cisco also recommends configuration of authentication and authorization together with accounting.

```
aaa accounting exec default start-stop group tacacs+
aaa accounting commands 0 default start-stop group tacacs+
aaa accounting commands 1 default start-stop group tacacs+
aaa accounting commands 15 default start-stop group tacacs+
```

For additional information about the implementation of AAA, see the section Authentication, Authorization, and Accounting section of Cisco Guide to Harden Cisco IOS Devices.

Use TACACS+ Authorization to Restrict Commands

Command authorization via TACACS+ should be enforced to keep tight control over commands that network administrators should not use without specific reasons. This can be accomplished by configuring authentication and command authorization via TACACS+.

The following example shows how to configure a Cisco IOS device for TACACS+ authentication, command authorization, and command accounting:

```
aaa new-model
aaa authentication login default group tacacs+ local enable
aaa authentication enable default group tacacs+ enable
aaa authorization config-commands
aaa authorization exec default group tacacs+ local if-authenticated
aaa authorization commands 1 default group tacacs+ local if-authenticated
aaa authorization commands 15 default group tacacs+ local if-authenticated
aaa accounting exec default start-stop group tacacs+
aaa accounting commands 0 default start-stop group tacacs+
aaa accounting commands 1 default start-stop group tacacs+
```

```
aaa accounting commands 15 default start-stop group tacacs+
aaa session-id common
```

When authorization is in place, the following commands should be restricted or prohibited by configuring the external AAA server. The following commands are particularly relevant to ensure that the Cisco IOS Software run-time memory and boot sequence are not modified:

- **gdb ***
- **test ***
- **tclsh ***
- **service internal**
- **config-register***
- **boot***
- **upgrade***

The following commands may be used to connect to line cards or switch processors on products that support them. They are particularly important because after the Cisco IOS device is connected to a line card or switch processor, the commands executed are not logged or authorized using the AAA server.

- **attach ***
- **remote ***
- **ipc-con ***
- **if-con ***
- **execute-on ***

The following commands may be used to show a particular state of the system. They are important because they can be used during a reconnaissance attack to study the system and prepare an attack using other commands:

- **show platform ***
- **show region**
- **show memory ***

Cisco IOS Software allows the use of the **do-exec <command>** in configuration mode. It is important that policies for authentication, command authorization, and command accounting take this feature into account by restricting or prohibiting any of the commands detailed in this section even when they are preceded by the **do-exec** command (for example, **do-exec test ***).

Implement Credentials Management

Passwords control access to resources or devices. A password or secret is defined and used to authenticate requests. When a request is received for access to a resource or device, the request is challenged for verification of the password and identity, and access can be granted, denied, or limited based on the result.

As a security best practice, passwords should be managed with a TACACS+ or RADIUS authentication server. However, a locally configured password for privileged access is still needed in the event of a failure of the

TACACS+ or RADIUS services. A device can also have other password information in its configuration, such as an NTP key, Simple Network Management Protocol (SNMP) community string, or routing protocol key.

The **enable secret** command is used to set the password that grants privileged administrative access to the Cisco IOS system. The **enable secret** command must be used, rather than the older **enable password** command. The **enable password** command uses a weak encryption algorithm.

If no enable secret is set and a password is configured for the console line, the console password can be used to receive privileged access, even from a remote vty session. This action is almost certainly unwanted and is another reason to ensure configuration of an enable secret.

The **service password-encryption** global configuration command directs Cisco IOS Software to encrypt the passwords, Challenge Handshake Authentication Protocol (CHAP) secrets, and similar data that are saved in its configuration file. Such encryption is useful to prevent casual observers from reading passwords, such as when they look at the screen over the shoulder of an administrator. However, the algorithm used by the **service password-encryption** command is a simple Vigenère cipher. The algorithm is not designed to protect configuration files against serious analysis by even slightly sophisticated attackers and must not be used for this purpose. Any Cisco IOS configuration file that contains encrypted passwords must be treated with the same care that is used for a cleartext list of those same passwords.

Although this weak encryption algorithm is not used by the **enable secret** command, it is used by the **enable password** global configuration command, as well as the **password** line configuration command. Passwords of this type must be eliminated and the **enable secret**command or the Enhanced Password Security feature needs to be used.

The **enable secret** command and the Enhanced Password Security feature use salted MD5 for password hashing. This algorithm has had considerable public review and is not known to be reversible. However, the algorithm is subject to dictionary attacks. In a dictionary attack, an attacker tries every word in a dictionary or other list of candidate passwords to find a match. Therefore, configuration files must be securely stored and only shared with trusted individuals.

Particular care should be taken in protecting network administrator credentials from theft because privileged access to the Cisco IOS device may be used to compromise the integrity of the memory, compromise the confidentiality of the data and configuration, and affect operations.

Cisco recommends the use of two-factor authentication for device management.

Implement Configuration Controls

Configuration management is a process by which configuration changes are proposed, reviewed, approved, and deployed. In the context of a Cisco IOS device configuration, two additional aspects of configuration management are critical: configuration archives and security.

You can use configuration archives to roll back changes that are made to network devices. In a security context, configuration archives can also be used to determine which security changes were made and when these changes

occurred. In conjunction with AAA log data, this information can assist in the security auditing of network devices.

The configuration of a Cisco IOS device contains many sensitive details. Usernames, passwords, and the contents of access control lists are examples of this type of information. The repository that you use to archive Cisco IOS device configurations needs to be secured. Insecure access to this information can undermine the security of the entire network.

Additional information is in the Cisco IOS Software Configuration Management section of Cisco Guide to Harden Cisco IOS Devices.

Protect Interactive Access to Devices

After AAA has been implemented to control which users can log in to particular network devices, access control should be implemented to limit IP addresses from which users may perform management functions on a network device. This access control includes multiple security features and solutions to limit access to a device:

- VTY access classes
- Management Plane Protection (MPP)
- Control Plane Policing (CoPP)
- Control Plane Protection (CPPr)
- Infrastructure access control lists (iACL)
- SNMP access lists

For more information, see the following sections of Cisco Guide to Harden Cisco IOS Devices: Secure Interactive Management Sessions and Fortify the Simple Network Management Protocol.

Many protocols carry sensitive network management data. You must use secure protocols whenever possible. A secure protocol choice includes the use of SSH instead of Telnet so that both authentication data and management information are encrypted. In addition, you must use secure file transfer protocols when you copy configuration data. An example is the use of SCP in place of FTP or TFTP.

See the Secure Interactive Management Sessions section of Cisco Guide to Harden Cisco IOS Devices for more information about the secure management of Cisco IOS devices.

Gain Traffic Visibility with NetFlow

NetFlow enables you to monitor traffic flows in the network. Originally intended to export traffic information to network management applications, NetFlow can also be used to show flow information on a router. This capability allows you to see what traffic traverses the network in real time. Regardless of whether flow information is exported to a remote collector, you are advised to configure network devices for NetFlow so that it can be used reactively if needed.

More information about this feature and using NetFlow to identify a possibly compromised device or network is in the Traffic Identification and Traceback section of Cisco Guide to Harden Cisco IOS Devices and in Telemetry-Based Infrastructure Device Integrity Monitoring.

Use Centralized and Comprehensive Logging

For network administrators to understand events taking place on a network, a comprehensive logging structure using centralized log collection and correlation must be implemented. Additionally, a standardized logging and time configuration must be deployed on all network devices to facilitate accurate logging. Furthermore, logging from the AAA functions in the network should be included in the centralized logging implementation.

After comprehensive logging is in place on a network, the collected data must be used to monitor network activity for events that may indicate unauthorized access to a network device or unauthorized actions by legitimate users. These types of events could represent the first step in undermining the security on a Cisco IOS device. Because the following items may represent unauthorized access or unauthorized actions, they should be monitored closely:

- The transmission of Cisco IOS Software images to a Cisco IOS device using the **copy** command or local SCP, TFTP, or FTP server functionality.
- The attempted execution of certain high-risk EXEC commands.
  The **copy**, **gdb**, **more**, **configure**, **tclsh,** and **test** commands are some examples of commands that should be monitored. This list is not exhaustive.
- Modification of the boot environment in use on the network devices. This specifically includes the **boot** and **config-register** global configuration commands.
- Modification of the security configuration for a Cisco IOS device. This may include the removal of VTY access classes or the logging configuration or the addition of new administrative users.
- Logging related to the insertion or removal of storage media, such as flash devices.
- SNMP-related logging of attempts to modify the Cisco IOS device configuration or perform file management tasks.
- The planned and unplanned reload of the Cisco IOS Software due to a software crash or the use of the **reload** command.

For more information, see the Centralize Log Collection and Monitoring and Logging Best Practices sections of Cisco Guide to Harden Cisco IOS Devices.

Conclusion

In conclusion, as interest in Cisco IOS Software integrity assurance is growing, this document presented various methods for an administrator to assess the integrity of the software running on his or her Cisco IOS device. These include image and memory verification, command checks, boot history checks, and more. Most miscreants targeting Cisco IOS router software would, in theory, attempt to achieve it by using hidden commands, compromising images, or exploiting vulnerabilities. These techniques can usually be prevented by implementing common recommendations and best practices that are published by Cisco and summarized in this document. Command authorization and accounting, logging, credential management, image signing, vulnerability control, and device hardening are some of the most important practices that will not only prevent Cisco IOS Software modification in nearly all cases, but will also ensure good security policy.

Acknowledgments

Authors:

- Panos Kampanakis (pkampana[at]cisco[dot]com) is a member of the Applied Security Intelligence team in the Security Intelligence Operations organization.
- Stefano De Crescenzo (sdecresc[at]cisco[dot]com) is a member of the PSIRT team in the Security Intelligence Operations organization.
- Xavier Brouckaert (xabrouck[at]cisco[dot]com) is a member of the PSIRT team in the Security Intelligence Operations organization.
- Dario Ciccarone (dciccaro[at]cisco[dot]com) is a member of the PSIRT team in the Security Intelligence Operations organization.

References

Cisco Guide to Harden Cisco IOS Devices
//www.cisco.com/c/en/us/support/docs/ip/access-lists/13608-21.html

Cisco IOS Image Verification
//www.cisco.com/web/about/security/intelligence/iosimage.html

Offline Analysis of IOS Image Integrity Blog
http://blogs.cisco.com/security/offline-analysis-of-ios-image-integrity/

Securing Tool Command Language on Cisco IOS
//www.cisco.com/web/about/security/intelligence/securetcl.html

Cisco Security Vulnerability Policy
//www.cisco.com/web/about/security/psirt/security_vulnerability_policy.html

Use of the Configuration Register on All Cisco Routers
//www.cisco.com/c/en/us/support/docs/routers/10000-series-routers/50421-config-register-use.html

Digitally Signed Cisco Software
//www.cisco.com/c/en/us/td/docs/ios-xml/ios/sys-image-mgmt/configuration/15-mt/sysimgmgmt-15-mt-book/sysimgmgmt-dgtly-sgnd-sw.html

Cisco IOS Software Checker
https://sec.cloudapps.cisco.com/security/center/selectIOSVersion.x

Creating Core Dumps
//www.cisco.com/en/US/docs/internetworking/troubleshooting/guide/tr19aa.html

Cisco IOS Configuration Guide
//www.cisco.com/c/en/us/support/ios-nx-os-software/ios-15-3m-t/products-installation-and-configuration-guides-list.html

MD5 File Validation
//www.cisco.com/c/en/us/td/docs/ios-xml/ios/sys-image-mgmt/configuration/15-mt/sysimgmgmt-15-mt-book/sysimgmgmt-md5.html

Image Verification
//www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_usr_cfg/configuration/15-mt/sec-usr-cfg-15-mt-book/sec-image-verifctn.html

Telemetry-Based Infrastructure Device Integrity Monitoring
//www.cisco.com/web/about/security/intelligence/network-integrity-monitoring.html

Cisco IOS XE Software Integrity Assurance
//www.cisco.com/web/about/security/intelligence/ios-xe-integrity-assurance.html

Cisco Value Chain Security
//www.cisco.com/web/about/doing_business/trust-center/built-in-security/supply-chain-security.html

Revision History

Date	Description
September 7, 2018	Moved content to a new URL.
June 8, 2015	Added additional information to the Introduction, Potential Attack Methods, Commands, Manipulating Cisco IOS Images, Vulnerabilities, and ROM Monitor sections. Added Cisco Secure Boot and Cisco Supply Chain Security sections. Added reference for Cisco Supply Chain Security.
May 29, 2015	Added caution statement in the "Text Memory Section Export" and "Verify MD5 Validation Feature for the Text Region" sections to indicate that certain commands should not be used with certain platforms.
November 6, 2014	Removed RADIUS from "Checking External Accounting Logs."
July 17, 2014	Added the sections "Verifying Authenticity for Digitally Signed Images" and "Checking That Cisco IOS Software Call Stacks Are Within the Text Section Boundaries." Added to the list of unusual and suspicious commands. Added links to *Telemetry-Based Infrastructure Device Integrity Monitoring* and *Cisco IOS XE Software Integrity Assurance*.
April 16, 2014	Initial public release.

Back to Top
