Network Device Integrity (NDI) on Cisco IOS Devices

February 23, 2016

Table of Contents

Network Device Integrity (NDI) on Cisco IOS Devices

Cisco IOS - System Information

Cisco Systems, Inc. produces numerous network devices on various platforms with differing architectures, ranging from routers and switches to firewalls. Even though some systems are capable of running the same Internetwork Operating System (IOS), the available functionality on different platforms can vary greatly. How to obtain and analyze some of the information necessary for performing Network Device Integrity (NDI) specifically for Cisco IOS systems is described here, but the commands shown may not apply to all IOS systems.

For all of the examples provided below, some of the output may have been removed for simplicity, so only the relevant output is shown. Also, a prompt that ends with ">" indicates that only user level access was required to run the command (privilege level 1), and a prompt that ends with "#" indicates that privileged level access was required to run the command (privilege level 15).

The **host name**, **model, device serial number**, **uptime**, **firmware version**, **boot settings**, **last reboot reason** and **running OS version** can be obtained from "show version" or "show hardware".

```
Router>show version
Cisco IOS Software, 2801 Software (C2801-ADVIPSERVICESK9-M), Version 12.4(21a), RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2008 by Cisco Systems, Inc.
Compiled Mon 29-Sep-08 16:31 by prod_rel_team

ROM: System Bootstrap, Version 12.4(13r)T, RELEASE SOFTWARE (fc1)

Router uptime is 20 hours, 38 minutes
System returned to ROM by power-on
System image file is "flash:c2801-advipservicesk9-mz.124-21a.bin"

Cisco 2801 (revision 7.0) with 352256K/40960K bytes of memory.
Processor board ID FTX13298045
2 FastEthernet interfaces
8 terminal lines
1 Virtual Private Network (VPN) Module
DRAM configuration is 64 bits wide with parity disabled.
191K bytes of NVRAM.
126000K bytes of ATA CompactFlash (Read/Write)

Configuration register is 0x2102
```

The configuration register is considered the **boot settings**, and is normally set to either 0x2102 or 0xF. The firmware on a Cisco IOS device is usually referred to as the ROM monitor (commonly shortened to ROMMON or just ROM). The **firmware version** is different from the IOS Software version, though it may have a similar format.

As shown below, switches may also provide additional information, such as the base ethernet **MAC address**, **additional serial numbers**, and a more detailed **model number**. If the switch is stackable, connected switches will also be listed. Additional information about stacked switches can be obtained with "show switch detail".

```
Switch>show version
Cisco IOS Software, C3750 Software (C3750-IPSERVICESK9-M), Version 12.2(55)SE3, RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2011 by Cisco Systems, Inc.
Compiled Thu 05-May-11 16:29 by prod_rel_team
Image text-base: 0x01000000, data-base: 0x02F00000

ROM: Bootstrap program is C3750 boot loader
BOOTLDR: C3750 Boot Loader (C3750-HBOOT-M) Version 12.2(44)SE5, RELEASE SOFTWARE (fc1)

Switch uptime is 2 minutes
System returned to ROM by power-on
System image file is "flash:/c3750-ipservicesk9-mz.122-55.SE3.bin"

cisco WS-C3750G-24T (PowerPC405) processor (revision L0) with 131072K bytes of memory.
Processor board ID FHK1016Y05U
Last reset from power-on
1 Virtual Ethernet interface
24 Gigabit Ethernet interfaces
The password-recovery mechanism is enabled.

512K bytes of flash-simulated non-volatile configuration memory.
Base ethernet MAC Address       : 00:12:7F:33:CC:80
Motherboard assembly number     : 73-8046-07
Power supply part number        : 341-0048-01
Motherboard serial number       : CAT084303ZG
Power supply serial number      : DTH08060KGN
Model revision number           : L0
Motherboard revision number     : A0
Model number                    : WS-C3750G-24T-S
System serial number            : FHK1016Y05U
Top Assembly Part Number        : 800-25855-01
Top Assembly Revision Number    : D0
Hardware Board Revision Number  : 0x05

Switch Ports Model              SW Version            SW Image
------ ----- -----              ----------            ----------
*    1 24    WS-C3750G-24T      12.2(55)SE3           C3750-IPSERVICESK9-M

Configuration register is 0xF
```

Additional serial numbers can be obtained from "show inventory raw". The **device serial number** (chassis) and a more accurate **model** number can also be obtained from this output, or "show inventory".

```
Router>show inventory raw
NAME: "chassis", DESCR: "2801 chassis"
PID: CISCO2801          , VID: V05 , SN: FTX13298045

NAME: "Chassis Slot", DESCR: "C2801 Chassis Slot"
PID:                    , VID:    , SN:

NAME: "motherboard", DESCR: "C2801 Motherboard with 2 Fast Ethernet"
PID:                    , VID: V05 , SN: FTX13298045

NAME: "Daughter card slot:0", DESCR: "C2801 DaughterCard Slot"
PID:                    , VID:    , SN:

NAME: "Daughter card slot:1", DESCR: "C2801 DaughterCard Slot"
PID:                    , VID:    , SN:

NAME: "WIC/VIC/HWIC 1", DESCR: "High Speed Wan Interface card with 8 RS232 async port(HWIC-8A)"
PID: HWIC-8A            , VID: V01 , SN: FOC1346364J
```

```
Switch>show inventory raw
NAME: "Cat37xx Stacking", DESCR: "Catalyst 37xx Switch Stack"
PID:                    , VID:      , SN:

NAME: "1", DESCR: "WS-C3750G-24T"
PID: WS-C3750G-24T-S    , VID: L0   , SN: FHK1016Y05U

NAME: "Switch 1 - WS-C3750G-24T - Power Supply 0", DESCR: "Switch 1 - WS-C3750G-24T - Power Supply 0"
PID:                    , VID:      , SN: DTH08060KGN

NAME: "Switch 1 - WS-C3750G-24T - Fan 0", DESCR: "Switch 1 - WS-C3750G-24T - Fan 0"
PID:                    , VID:      , SN:
```

Additional serial numbers can also be obtained from "show diag", as well as the **chassis MAC address**, the **device serial number** (chassis) and a more detailed **model** number. This command may not be valid on all systems. Additional hardware information can be obtained with "show idprom all", "show module" and "show diagbus" on supported systems.

```
Router>show diag
Slot 0:
        C2801 2FE 4SLOT Mainboard Port adapter, 10 ports
        Port adapter is analyzed
        Port adapter insertion time 04:30:48 ago
        EEPROM contents at hardware discovery:
        Chassis MAC Address      : 0025.454d.8524
        MAC Address block size   : 34
        PCB Serial Number        : FOC13263GD5
        Hardware Revision        : 7.0
        Part Number              : 73-8190-08
        Board Revision           : D0
        Top Assy. Part Number    : 800-23435-06
        Deviation Number         : 100870
        Fab Version              : 04
        CLEI Code                : IPMK400ARA
        RMA Test History         : 00
        RMA Number               : 0-0-0-0
        RMA History              : 00
        Product (FRU) Number     : CISCO2801
        Version Identifier       : V05
        Processor type           : 86
        Chassis Serial Number    : FTX13298045
        EEPROM format version 4
        EEPROM contents (hex):

        WIC/VIC/HWIC Slot 1:
        HWIC Serial 8A daughter card
        Hardware Revision        : 1.0
        Part Number              : 73-8973-05
        Board Revision           : B0
        Deviation Number         : 0
        Fab Version              : 05
        PCB Serial Number        : FOC1346364J
        RMA Test History         : 00
        RMA Number               : 0-0-0-0
        RMA History              : 00
        Top Assy. Part Number    : 800-23793-01
        Connector Type           : 01
        Product (FRU) Number     : HWIC-8A
        Version Identifier       : V01
        CLEI Code                : CNUIADCAAA
        EEPROM format version 4
        EEPROM contents (hex):
```

The installed **firmware versions** can also be obtained from the output of "show rom-monitor". Routers generally have two regions to store a copy of the firmware. The first is a ReadOnly region that cannot be modified, and is hardcoded when the device was manufactured. The second is an Upgrade region that allows the firmware to be upgraded. The firmware stored in the Upgrade region can be used when it exists, otherwise the firmware stored in the ReadOnly region will be used to boot the device. Only the currently used version is displayed in "show version" above. It appears that the firmware of switches can only be upgraded through ROMMON, so the "show rom-monitor" command only applies to routers.

```
Router>show rom-monitor
ReadOnly 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.

No upgrade ROMMON programmed or not yet run
Currently running ROMMON from ReadOnly region
ROMMON from ReadOnly region is selected for next boot
```

The following two examples[1] show when the firmware has been upgraded on a router. The first example shows a router that has not yet been rebooted since the upgrade, and the second is a router that has already been rebooted after the upgrade. The firmware of a Cisco IOS router can be upgraded with "upgrade rom-monitor file". The ReadOnly region can also be selected as a boot preference over the Upgrade region with "upgrade rom-monitor preference".

```
Router>show rom-monitor
ReadOnly ROMMON version:

System Bootstrap, Version 12.3(8r)YH3, RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 2005 by cisco Systems, Inc.

Upgrade ROMMON version is not visible due to recent license activity,
such as license installation, removal, or the use of evaluation license
Reload is required to show the upgrade ROMMON version

Currently running ROMMON from Upgrade region
ROMMON from Upgrade region is selected for next boot
```

```
Router>show rom-monitor
ReadOnly ROMMON version:

System Bootstrap, Version 12.2(20031011:151758)
Copyright (c) 2004 by Cisco Systems, Inc.

Upgrade ROMMON version:

System Bootstrap, Version 12.2(20031011:151758)
Copyright (c) 2004 by Cisco Systems, Inc.

Currently running ROMMON from Upgrade region
ROMMON from Upgrade region is selected for next boot
```

The **unique attribute** can be obtained by computing a cryptographic hash of the startup-config with "verify /md5 nvram:/startup-config". This can be used to uniquely identify a system from other systems on the network without relying on the hardware. It is highly unlikely that two different systems would have the exact same configuration. The startup-config is chosen over the running-config because some systems contain dynamic information in the running-config. Even though newer operating systems may support additional hash types (such as /sha1), the same cryptographic algorithm should be used for all systems of the same type throughout the entire network so they can be consistently compared.

```
Router#verify /md5 nvram:/startup-config
.Done!
verify /md5 (nvram:/startup-config) = d272f622726e3e78fa4be5420a223900
```

A list of **IP addresses** assigned to the device can be obtained from "show running-config", "show interfaces", "show ip interface", "show ip interface brief", or "show protocols". A short list of IP addresses can be obtained with "show ip interface brief", but it may be preferred to utilize "show interfaces" since it provides detailed information along with the **MAC addresses**, including the "burned in address" (bia). It is common for MAC addresses to be incremental on the same piece of equipment as shown here. A list of IP addresses is useful when correlating other information, such as events from log messages.

```
Router>show interfaces
FastEthernet0/0 is up, line protocol is up
  Hardware is Gt96k FE, address is 0025.454d.8524 (bia 0025.454d.8524)
  Internet address is 10.0.0.1/24

FastEthernet0/1 is up, line protocol is down
  Hardware is Gt96k FE, address is 0025.454d.8525 (bia 0025.454d.8525)
  Internet address is 10.0.1.1/24
```

Unfortunately, "show interfaces" does not provide helper **IP addresses** (utilized by the Hot Standby Router Protocol, HSRP) or secondary **IP addresses**, which can be obtained with "show ip interface", or "show running-config".

```
Router>show ip interface
FastEthernet0/0 is up, line protocol is up
  Internet address is 10.0.0.1/24
  Broadcast address is 255.255.255.255
  Address determined by non-volatile memory
  MTU is 1500 bytes
  Helper address is 10.0.0.254
  Directed broadcast forwarding is disabled
  Secondary address 10.0.0.2/24
```

```
Router#show running-config
interface FastEthernet0/0
 ip address 10.0.0.2 255.255.255.0 secondary
 ip address 10.0.0.1 255.255.255.0
 ip helper-address 10.0.0.254
 duplex auto
 speed auto
 ipv6 address 2001::1/64
 ipv6 enable
```

Both "show interfaces" and "show ip interface" do not provide all **IPv6 addresses**, which can instead be obtained with "show ipv6 interface". IPv6 link-local addresses are automatically configured on IPv6 enabled interfaces and are not included in "show ip interface" or "show running-config" as shown above.

```
Router>show ipv6 interface
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::225:45FF:FE4D:8524 [TEN]
  Global unicast address(es):
    2001::1, subnet is 2001::/64 [TEN]
```

Cisco IOS systems support a list of **boot OS versions** to attempt to load whenever the device is booted. If an image file in the list does not exist or cannot be loaded, it will automatically attempt to load the next image file in the list. If there are no more image files in the list, or the list is empty, it will search the flash for the first bootable image. The order of this list can be changed as desired.

Where this list is defined depends on the type and version of the device and operating system. Examine the lines between "boot-start-marker" and "boot-end-marker" in "show startup-config". This list will also exist in the running-config, but the startup-config is read when the device first boots, which may differ from the running-config if it has not been saved. Multiple entries can exist between the markers, and the list is empty if nothing is between the markers. Older systems may simply specify the list in the configuration without the markers. It may also be necessary to examine and research other lines in the configuration that start with the "boot" keyword, such as "boot bootldr".

```
Router#show startup-config
boot-start-marker
boot system flash flash:/c2801-advipservicesk9-mz.124-21a.bin
boot-end-marker
```

The **boot OS versions** may also be listed on some systems with "show boot" or "show bootvar". Multiple entries are separated by a semicolon. Individual file names may include a comma and a number at the end, as shown, and is not actually part of the file name.

```
Switch#show boot
BOOT path-list      : flash:c3750-ipbase-mz.122-25.SEB4/c3750-ipbase-mz.122-25.SEB4.bin
Config file         : flash:/config.text
Private Config file : flash:/private-config.text
Enable Break        : no
Manual Boot         : no
```

```
Switch#show bootvar
BOOT variable = sup-bootflash:s72033-ipservicesk9_wan-mz.122-33.SXJ1.bin,1;
CONFIG_FILE variable does not exist
BOOTLDR variable does not exist
Configuration register is 0x2102
```

Even though a file name may be specified in the list above, it does not necessarily mean the file actually exists. The existence of a file can be determined with the "dir" command. Any **other OS versions** or executable files that are stored on the file system should also be recorded, especially if they are not currently in use or defined in the boot list. This can also help determine what the **boot OS version** would be if the boot list is empty. The commands "dir all-filesystems", "dir ." and "show flash" can be used to list the initial root file systems available. Some system may require the use of "show bootflash" or "show slot0". The "show file systems" command can also be used to list all of the available file systems. It may be necessary to traverse each file system recursively to look for files in other directories (indicated with a "d" in the permissions), which can be performed with additional "dir" commands with the full path as the only parameter. The "/recursive" parameter may also be useful for this purpose.

```
Router#dir all-filesystems
Directory of system:/

    2  drwx           0                   <no date>  its
  115  dr-x           0                   <no date>  lib
  145  dr-x           0                   <no date>  memory
    1  -rw-        1476                   <no date>  running-config
  114  dr-x           0                   <no date>  vfiles

Directory of nvram:/

  185  -rw-        1476                   <no date>  startup-config
  186  ----        3582                   <no date>  private-config
  187  -rw-        1476                   <no date>  underlying-config
    1  -rw-           0                   <no date>  ifIndex-table

Directory of flash:/

    1  -rw-    30848136  Jul 13 2009 16:47:00 +00:00  c2801-advipservicesk9-mz.124-21a.bin
    2  -rw-        2746  Jul 13 2009 17:41:42 +00:00  sdmconfig-2801.cfg
    3  -rw-      931840  Jul 13 2009 17:42:02 +00:00  es.tar
```

```
Router#dir .
Directory of flash:/

    1  -rw-    30848136  Jul 13 2009 16:47:00 +00:00  c2801-advipservicesk9-mz.124-21a.bin
    2  -rw-        2746  Jul 13 2009 17:41:42 +00:00  sdmconfig-2801.cfg
    3  -rw-      931840  Jul 13 2009 17:42:02 +00:00  es.tar
    4  -rw-     1505280  Jul 13 2009 17:42:26 +00:00  common.tar
    5  -rw-        1038  Jul 13 2009 17:42:46 +00:00  home.shtml
    6  -rw-      112640  Jul 13 2009 17:43:04 +00:00  home.tar
    7  -rw-      527849  Jul 13 2009 17:43:26 +00:00  128MB.sdf
    8  -rw-     1697952  Jul 13 2009 17:44:00 +00:00  securedesktop-ios-3.1.1.45-k9.pkg
    9  -rw-      415956  Jul 13 2009 17:44:30 +00:00  sslclient-win-1.1.4.176.pkg
```

The "file verify auto" global configuration command will enable the Cisco IOS Image Verification Feature[2] [3]. This feature will execute the "verify" command on all images when the "copy" or "reload" commands are used. The existence of this **self-verification** feature can be confirmed with "show running-config".

```
Router#show running-config

file verify auto
```

Most configuration information can be obtained from "show running-config". Some systems will support the command "show running-config all" which will provide all of the default configuration values versus only those values that have been changed from the default. This information can be useful when attempting to determine what a default configuration may be set to.

Cisco IOS - Unauthorized Access Detection

To successfully detect unauthorized access, logging must be properly enabled across the network to ensure all actions and events across multiple systems are properly recorded. The following criteria should be followed when configuring logging on a Cisco IOS system. A list of example global configuration commands is provided below for reference.

- Syslog logging should be set to at least informational (6). The higher debugging (7) setting may generate too many log messages to manage.
- Authentication successes and failures should always be logged.
- All log messages should be sent to at least two remote log servers to ensure they are not lost or destroyed. Connectivity to the log servers should be verified to ensure messages are received.
- The logging buffered (stored in memory) can be convenient for accessing recent logs. The buffer should be large enough to prevent older messages from rolling over too quickly.
- Console logging may need to be disabled to ensure local log messages are stored in the buffer and not only written to the console.
- Timestamps should include at least the date, time and time zone to ensure log messages can be correlated with other events. The milliseconds can provide additional granularity, and the year can be useful when log messages are not appropriately archived.
- The clock should be synchronized with at least two external time sources to ensure log messages can be properly correlated with events from other systems. This can be achieved with the proper configuration of the Network Time Protocol (NTP).
- Time source (NTP) authentication should be enabled to prevent an adversary from spoofing external time updates which can be used to disguise when an event occurred.
- Logging may also need to be enabled for specific events, such as access list permits and denies, routing updates, or utilization of specific services.
- Accounting should also be enabled when a centralized Authentication, Authorization and Accounting (AAA) server is utilized. This will provide a mechanism for tracking specific commands that were executed by individual accounts and events that occurred on the system, or when unauthorized actions were attempted.

```
logging buffered 65535 informational
login on-failure log
login on-success log
logging 10.0.5.14
logging 10.0.7.14
no logging console
service timestamps log datetime msec show-timezone year

ntp server 10.0.5.123
ntp server 10.0.7.123
ntp authenticate
ntp authentication-key 1 md5 NTP_PASSWORD

aaa accounting exec default start-stop group tacacs+
aaa accounting commands 1 default start-stop group tacacs+
aaa accounting commands 15 default start-stop group tacacs+
aaa accounting network default start-stop group tacacs+
aaa accounting connection default start-stop group tacacs+
aaa accounting system default start-stop group tacacs+
aaa accounting resource default start-stop group tacacs+
```

Significant information can be obtained from the output of "show logging" concerning potential unauthorized access to a device. All log messages that are not understood or do not correlate to known administrative activity should warrant an investigation. Significant information is available on cisco.com and online support forums concerning the numerous log messages a Cisco IOS device can generate.

The time of an event should be included with the log messages when timestamps are properly configured. If the system clock has not been set, the date and time are preceded by an asterisk (*) to indicate that the date and time are probably not correct. If the system clock has been set but is currently unsynchronized, the date and time are preceded by a period (.) to indicate that the date and time may not be accurate. Several example log messages from the output of "show logging" have been provided in each of the scenarios described in this section, but the timestamps have been removed for simplicity

```
Switch#show logging
*Mar 20 20:39:34.412 UTC: %SYS-5-CONFIG_I: Configured from console by admin on vty0 (10.0.0.2)
```

It is crucial to note that all of the log messages provided here are simply examples of specific types of events. The existence of any of these log messages or events does not necessarily mean that a compromise has occurred. All suspicious log messages should be investigated to determine the cause, which may actually be benign depending on the network where the event occurred. An event can usually only be declared malicious after correlating multiple log messages together. If a behavior listed in one of the below log messages is normal and expected for a given network, it can likely be correlated with an authorized action or individual.

Login Access

It is critical to know what accounts were used to login to a particular device and when any login attempts occurred, including authentication failures. Any abnormal login attempts could be an attempt to gain unauthorized access. The following log messages will be stored in the local buffer when the "login on-failure log" and "login on-success log" global configuration commands are enabled. Otherwise the accounting logs may be the only source of authentication successes and failures.

```
%SEC_LOGIN-5-LOGIN_SUCCESS: Login Success [user: cisco] [Source: 192.168.0.128] [localport: 22] at 00:58:07
UTC Mon Mar 1 1993
%SSH-5-SSH2_USERAUTH: User 'cisco' authentication for SSH2 Session from 192.168.0.128 (tty = 0) using crypto
cipher 'aes128-cbc', hmac 'hmac-md5' Succeeded

%SEC_LOGIN-4-LOGIN_FAILED: Login failed [user: hacker] [Source: 192.168.0.130] [localport: 22] [Reason:
Login Authentication Failed] at 00:59:13 UTC Mon Mar 1 1993
```

Configuration Changes

The detection of any changes to a device's configuration can instantly be an indication of unauthorized access. Changes can be detected by examining the logs for any instances when the global configuration mode was utilized. Unfortunately, there is not an easy mechanism to determine what change was actually made without comparing the current configuration with a previous version, or by examining the accounting logs from a centralized AAA server, when available. Any configuration changes using non-standard methods, such as SNMP, TFTP, SCP, or via the console, could be an indication of unauthorized access.

```
%SYS-5-CONFIG_I: Configured from console by admin on vty0 (10.0.0.2)
%SYS-5-CONFIG_I: Configured from console by cisco on vty0 (192.168.0.128)
%SYS-5-CONFIG_I: Configured from console by cisco on console

%SYS-5-CONFIG_I: Configured from 10.2.43.161 by snmp
%SYS-5-CONFIG_I: Configured from tftp://10.23.6.13/config.txt by autouser on vty0 (10.42.1.6)
%SYS-5-CONFIG_I: Configured from scp://user@192.168.0.99/file by useracct on vty1 (192.168.0.130)
```

Some devices will save comments in the stored configuration that include information about the last time a configuration change was made, and when the stored configuration was saved to permanent storage. This can be viewed with "show startup-config". Any changes made by a user account not normally utilized for configuration changes could be an indication of unauthorized access.

```
Switch#show startup-config
Using 7560 out of 524288 bytes
!
! Last configuration change at 19:06:15 UTC Thu Oct 9 2015 by admin
! NVRAM config last updated at 19:06:17 UTC Thu Oct 9 2015 by admin
!
```

Other unusual log messages relating to the configuration may be an indication of unauthorized access. The configuration may be configured from memory when a device first boots, but usually should not be seen after the device has already booted. If a malicious user attempts to load additional lines into the configuration through other sources, an "unexpected end of configuration file" might be seen.

```
%SYS-5-CONFIG_I: Configured from memory by console

%PARSER-4-BADCFG: Unexpected end of configuration file.
```

Any configuration changes with how the system performs authentication or the creation of new user accounts could be an attempt to circumvent the authentication mechanisms to gain unauthorized access. Several of the errors listed below could be due to an administrative configuration error, which should be corrected to ensure the authentication configuration is correct and cannot be inadvertently bypassed.

```
%AAA-3-BAD-SERVERTYPEERROR: Cannot process account server type *invalid_group_handle*
%AAAA-4-BADMETHNAME: Bad authentication method-list name "local" (this is only a warning)
%AAA-3-BADSERVERTYPEERROR: Cannot process authentication server type tacacs+ (UNKNOWN)
%AAAA-4-SERVUNDEF: The server-group "MYGROUP" is not defined. Please define it.
```

Additionally, any configuration changes relating to how log messages are generated, stored or sent to external log servers could also be an indication of unauthorized access.

If the output of "show logging" is empty, or log messages appear to be missing, it is possible that an adversary may have cleared the logs with the command "clear logging". If there appears to be a significant number of useless log messages, an adversary may have attempted to fill up the log buffer to ensure older log messages were removed. In these cases, it will be necessary to examine the logs stored on the remote log servers.

Certain protocols are constantly synchronizing with other systems, such as routing protocols or the Network Time Protocol (NTP) for synchronizing the system clock. Successfully synchronizing with other systems that are not normally utilized could be the result of an adversary attempting to spoof an existing

system or to inject malicious or bogus information such as different routes or timestamps. The current status of NTP can be obtained with "show ntp status" and "show ntp associations".

```
%OSPF-5-ADJCHG: Process 100, Nbr 10.100.101.1 on Vlan8 from LOADING TO FULL, Loading Done

%NTP-5-PEERSYNC: NTP synced to peer 10.92.2.4
```

```
Switch#show ntp status
Clock is synchronized, stratum 12, reference is 192.168.0.1
nominal freq is 119.2092 Hz, actual freq is 119.2245 Hz, precision is 2**18
reference time is DA300EC3.BBCE167C (20:29:55.733 UTC Thu Dec 31 2015)
clock offset is -1.4960 msec, root delay is 3.72 msec
root dispersion is 13.55 msec, peer dispersion is 1.04 msec

Switch#show ntp associations

      address         ref clock     st  when  poll reach  delay  offset    disp
*~192.168.0.1      127.127.1.0      11   189   256  377     3.7   -1.50     1.0
 * master (synced), # master (unsynced), + selected, - candidate, ~ configured
```

Any log messages indicating an invalid signature while verifying operating system image files could be the result of an unauthorized software modification.

```
%SIGNATURE-4-NOT_PRESENT: %WARNING: Signature not found in file flash:/c850-advsecurityk9-mz.124-15.T16.bin
```

There exist several high risk activities that should generally never be utilized on Cisco IOS systems during normal operation, and these activities should be monitored through the centralized accounting server. The following commands are considered unusual and suspicious commands by Cisco[4], even though some of these commands can also be used when investigating a potential compromise.

gdb *	boot*	execute-on *
test *	upgrade*	service-monitor *
tclsh *	attach *	show region
debug *	remote *	show memory *
service internal	ipc-con *	show platform *
config-register*	if-con *	do-exec *(any of the above)*

GDB is an embedded debugger built into Cisco IOS and can be accessed with the "gdb" command. This command can potentially be utilized to modify memory and CPU registers while the system is running[5], potentially leading to a memory only implant. Any use of this command during normal operations should be investigated immediately. The "gdb" command does not show up as a valid command but can be used when provided with the correct parameters.

Cisco IOS also supports Tool Command Language (Tcl) scripting capabilities. Under normal operations, these scripting capabilities should not be utilized. Any use of the "tclsh" command could be an indication of unauthorized access or a rootkit. For supported systems, signed TCL support[6] can be enabled to prevent the execution of any Tcl scripts that are not properly signed.

Even though the "copy" commands are necessary for performing integrity checks and detecting software modifications of the operating system image files on Cisco IOS systems, the use of these commands are not necessary to support daily operations. The "copy" commands should only be utilized when upgrading or verifying the operating system image files stored on a device. Any use of these commands should be confirmed with the network administrators to ensure they were authorized.

Interface Changes

An interface that has changed state, either up or down, could indicate that an unauthorized system was connected or disconnected from the network. Unfortunately, there can be high false positive rate with this event if the culture of the network allows systems to be frequently turned on or off, or connected and disconnected from the network, or re-connected to different areas of the network. An interface change could also be an indication that a connected device was recently rebooted.

```
%LINK-3-UPDOWN: Interface GigabitEthernet1/0/20, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet1/0/20, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet1/0/20, changed state to down
%LINK-3-UPDOWN: Interface GigabitEthernet1/0/20, changed state to down

%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to administratively up
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to administratively down
```

The following log messages show when the interface counters have been cleared with "clear counters" on all of the interfaces or just a specific interface. Administrators generally do not have a reason to clear the counters, and thus this could be an indication of unauthorized access to hide network activity.

```
%CLEAR-5-COUNTERS: Clear counter on all interfaces by cisco on vty0 (192.168.0.128)
%CLEAR-5-COUNTERS: Clear counter on interface FastEthernet0/1 by cisco on vty0 (192.168.0.128)
```

The Media Access Control (MAC) addresses of neighboring systems can be obtained with "show arp" or "show ip arp". The MAC address table of a switch can be obtained with "show mac address-table" (or "show mac-address-table" on some systems). This is different than reviewing the MAC addresses assigned to the network device itself. Connected MAC addresses can be useful for correlating what systems are physically connected to the network device and can assist when determining if an unauthorized system was connected to the network or if systems have moved to a different location.

```
Switch>show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  192.168.0.34           0   0025.454d.8524  ARPA   Vlan1
Internet  192.168.0.33           -   0012.7f33.ccc0  ARPA   Vlan1
Internet  192.168.0.1            0   001b.217c.da10  ARPA   Vlan1
Internet  192.168.0.129          0   d067.e533.0e1f  ARPA   Vlan1

Switch>show mac address-table

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    001b.217c.da10    DYNAMIC     Gi1/0/1
   1    0025.454d.8524    DYNAMIC     Gi1/0/2
   1    d067.e533.0e1f    DYNAMIC     Gi1/0/3
```

Physical Access

Any events that occur on the console port could be an indication of unauthorized physical access. This assumes the device is normally accessed by remote administration services and not via physical access on the console port.

```
%SYS-5-CONFIG_I: Configured from console by cisco on console
```

Some systems may provide some basic usage statistics of the console port with the command "show line con 0". The "totalout" and "totalin" values will increase as data is sent or received on the console port. If the console port has not been utilized since the device last rebooted, these values should be zero, or only a very small number for "totalout" to account for the status information that is sent to the console port while the device booted. All lines can be listed with "show line", and other individual lines can be examined by providing additional parameters to the "show line" command, such as "aux 0" or "vty 0", to look for other inconsistencies. The "show terminal" command will provide information about the line used by the current session.

```
Router>show line con 0
   Tty Line Typ     Tx/Rx    A Modem  Roty AccO AccI  Uses  Noise Overruns  Int
     0    0 CTY              -    -      -    -    -     0      2    0/0      -

Line 0, Location: "", Type: ""
Length: 24 lines, Width: 80 columns
Baud rate (TX/RX) is 9600/9600, no parity, 2 stopbits, 8 databits
Status: Ready
Capabilities: none
Modem state: Ready
Modem hardware state: CTS* noDSR  DTR RTS
 TTY NUMBER 0
Receive Overrun = 0, Carrier Detect Loss = 0
Receive Clock Glitch = 0, Transmit Underrun = 0
CTS Loss = 0, Transmit Clock Glitch = 0
Outcount = 0 totalout = 822789 incount = 0 totalin = 4682
```

The firmware of a Cisco IOS device utilizes the configuration register to obtain the **boot settings** to determine how the device should be booted. This register can be changed to utilize the password recovery feature or to bypass loading the stored configuration when the device is booted. The current configuration register setting can be obtained from "show version" and is normally set to either 0x2102 or 0xF, depending on the type of device.

```
Router>show version
Configuration register is 0x2102
```

```
Switch>show version
Configuration register is 0xF
```

An adversary with physical access to a device may need to change the configuration register to gain unauthorized access to the system over the console port. The configuration register can be changed on some systems with the "config-register" global configuration command, or by breaking into ROMMON before the device boots. Any variances with the configuration register could be an indication of unauthorized access on the console port. A sophisticated adversary may know to change the configuration register back to its original setting, but this can also be detected prior to another reload of the device.

```
Router>show version
Configuration register is 0x2142 (will be 0x2102 at next reload)
```

Several Cisco IOS systems have Universal Serial Bus (USB) ports that enable external devices to be connected. Physical access is required to connect an external device via a USB port. The "show usb port", "show usb tree" and "show usb device" commands can be used to determine if any external USB devices

are currently connected to the system. The "show usb controllers" command can also be useful for obtaining information about the available USB ports on the device.

```
Router>show usb port
Port Number: 0
Status: Disabled
Connection State: Disconnected
Speed: Full
Power State: ON

Router>show usb tree
[Host Id: 1, Host Type: 1362HCD, Number of RH-Port: 1]
<Root Port0: Power=ON      Current State=Disabled>

Router>show usb device
Device not found
```

Many Cisco IOS systems utilize removable Compact Flash (CF) cards to store the operating system image files or configuration files. The following log messages indicate that the CF card has been removed and replaced, indicating that the device was physically accessed.

```
%FILESYS-5-CF: External CompactFlash removed

%FILESYS-5-CF: External CompactFlash inserted
```

The "show flash: filesys" command can be utilized on supported systems to list basic information about the CF card currently inserted, or to determine when a CF card is not present.

```
Router>show flash: filesys
******** ATA Flash Card Geometry/Format Info ********

ATA CARD GEOMETRY
   Number of Heads:       16
   Number of Cylinders    250
   Sectors per Cylinder   63
   Sector Size            512
   Total Sectors          252000

ATA CARD FORMAT
   Number of FAT Sectors  123
   Sectors Per Cluster    8
   Number of Clusters     31425
   Number of Data Sectors 251809
   Base Root Sector       374
   Base FAT Sector        128
   Base Data Sector       406

ATA MONLIB INFO
   Image Monlib size = 61080
   Disk monlib size = 65536
   Name = piptom-atafslib-m
   Monlib Start sector = 2
   Monlib End sector = 121
   Monlib updated by = C2801-ADVIPSERVICESK9-M12.4(21a)
   Monlib version = 1

Router>show flash: filesys
%Error show flash: (No such device)
```

Unscheduled Reboots

An unscheduled reboot could be an indication of unauthorized access, an attempt to gain unauthorized access, or a potential software modification of the operating system image files. The following log messages indicate that a system was rebooted or that a system crashed.

```
%SYS-5-RESTART: System restarted --
%SYS-6-BOOTTIME: Time taken to reboot after reload =   91 seconds

%PLATFORM-1-CRASHED:
%PLATFORM-1-CRASHED: Debug Exception (Could be NULL pointer dereference) Exception (0x2000)!
```

The output of "show version" should indicate how the device last rebooted. A device that returned to ROM by "power-on" was simply turned, which means the system may have been physically accessed.

```
Router>show version
Router uptime is 1 week, 2 hours, 25 minutes
System returned to ROM by power-on
```

A device that returned to ROM by "reload" was purposefully rebooted with the "reload" command. The time when a reload occurred, or the calculated difference from the uptime, should be correlated with other log messages to determine what account was used to perform the reload and what other actions may have occurred just before.

```
Router>show version
Router uptime is 13 minutes
System returned to ROM by reload at 17:24:19 UTC Wed Dec 30 2015
```

In the case shown below, the device had just crashed due to an error. This could have been an adversary attempting to exploit a vulnerability, or due to a software flaw. The output of "show context" may also be useful when investigating a recent crash on supported systems.

```
Switch>show version
Switch uptime is 4 minutes
System returned to ROM by address error at PC 0x2286F7C, address 0x0
System restarted at 18:56:38 UTC Mon Dec 28 2015
```

When a device crashes, several "crashinfo" directories and files may be created. These files can be examined to gather information about how or why the system crashed.

```
Switch#dir
Directory of flash:/

    8  drwx          64  Dec 28 2015 18:28:11 +00:00  crashinfo_ext
   11  drwx         128   Mar 1 1993 00:00:44 +00:00  crashinfo

Switch#dir flash:/crashinfo_ext
Directory of flash:/crashinfo_ext/

   10  -rwx      263706  Dec 28 2015 18:28:16 +00:00  crashinfo_ext_1
    7  -rwx      274729  Dec 28 2015 18:55:18 +00:00  crashinfo_ext_2

Switch#dir flash:/crashinfo/
Directory of flash:/crashinfo/

   12  -rwx        1943   Mar 1 1993 00:00:45 +00:00  crashinfo_1
   14  -rwx        1943   Mar 1 1993 00:00:44 +00:00  crashinfo_2
```

Unfortunately, it may be difficult to correlate the crashinfo files when time synchronization is not properly enabled. The time was synchronized in the above example so it could be determined that the file "flash:/crashinfo_ext/crashinfo_ext_2" corresponds to the last reboot of the device because of the time

when the system restarted. The contents of this file indicates that the last action performed prior to the crash was the command "verify /md5 system:/memory/main", which is known to crash switches.

```
Switch#more flash:/crashinfo_ext/crashinfo_ext_2

CMD: 'verify /md5 system:/memory/main' 18:55:13 UTC Mon Dec 28 2015
```

On some systems, "show history all" can be used to obtain similar command history. Unfortunately, "show history" only shows the command history from the current session.

```
Switch#show history all
*Dec 29 2015 20:34:52.142 UTC: %SSH-5-SSH2_SESSION: SSH2 Session request from 192.168.0.130 (tty = 0) using
crypto cipher 'aes256-cbc', hmac 'hmac-sha1' Succeeded
*Dec 29 2015 20:34:53.954 UTC: %SEC_LOGIN-5-LOGIN_SUCCESS: Login Success [user: cisco] [Source:
192.168.0.130] [localport: 22] at 20:34:53 UTC Tue Dec 29 2015
*Dec 29 2015 20:34:53.954 UTC: %SSH-5-SSH2_USERAUTH: User 'cisco' authentication for SSH2 Session from
192.168.0.130 (tty = 0) using crypto cipher 'aes256-cbc', hmac 'hmac-sha1' SucceededCMD: 'cisco' 20:34:56
UTC Tue Dec 29 2015
CMD: 'connect cisco' 20:34:56 UTC Tue Dec 29 2015
CMD: 'show run' 20:35:01 UTC Tue Dec 29 2015
```

In addition to unauthorized access, an unscheduled reboot could be the indication of an unauthorized software modification. It may be necessary to review the logs to determine if a different version of the operating system was set to boot or if a different version was copied to the device prior the last reboot.

Software Management

Obtaining the **running OS version**, **boot OS versions** and **other OS versions** were previously described under the section for obtaining system information. For most Cisco IOS systems, the running OS version and boot OS versions should be the same, and no other OS versions should be stored on the system. If this is not the case, it may be necessary to verify with network administrators why the versions differ, or to look for other indications of unauthorized access or software modifications.

The file timestamps of image files can be examined with the "dir" command to determine when they were copied to the device. File timestamps can be modified, so unfortunately this does not provide a high level of confidence. Log messages stored on the centralized log or accounting servers may be more accurate at determining the actual time when a file was copied to the device, if available.

```
Router#dir flash:c2801-advipservicesk9-mz.124-21a.bin
Directory of flash:/c2801-advipservicesk9-mz.124-21a.bin

    1  -rw-    30848136  Dec 22 2015 21:02:22 +00:00  c2801-advipservicesk9-mz.124-21a.bin
```

The Cisco IOS Resilient Configuration feature[7] enables a router to secure and maintain a working copy of the running image and configuration so that those files can withstand malicious attempts to erase the contents of persistent storage (NVRAM and flash). This feature can be enabled with the global configuration commands "secure boot-image" and "secure boot-config", which can only be applied or disabled when connected via the console port. Secured files will not appear on the output of a "dir" command because the file system prevents it. ROM monitor (ROMMON) mode does not have any such restriction and can be used to list and boot secured files. Use the "show secure bootset" command to verify archive existence.

```
Router#show secure bootset
IOS resilience router id FTX13298045

IOS image resilience version 12.4 activated at 21:08:23 UTC Mon Dec 21 2015
Secure archive flash:c2801-advipservicesk9-mz.124-21a.bin type is image (elf) []
  file size is 30848136 bytes, run size is 31013820 bytes
  Runnable image, entry point 0x8000F000, run from ram

IOS configuration resilience version 12.4 activated at 21:08:31 UTC Mon Dec 21 2015
Secure archive flash:.runcfg-20151221-210831.ar type is config
configuration archive size 1794 bytes
```

Unfortunately, an image or configuration file that has been secured with the Cisco IOS Resilient Configuration feature cannot be verified or copied. The system will report "No such file" or "File not found" when attempting to use the "verify" or "copy" commands, as shown below.

```
Router#dir flash:c2801-advipservicesk9-mz.124-21a.bin
Directory of flash:/c2801-advipservicesk9-mz.124-21a.bin

No such file

128716800 bytes total (30937088 bytes free)
```

Any changes to the configuration of the Cisco IOS Resilient Configuration feature could be indication that an unauthorized individual with privileged level access attempted to modify the boot image or the configuration.

```
*Dec 21 21:25:40.940 UTC: %IOS_RESILIENCE-5-IMAGE_RESIL_INACTIVE: Disabled secure image archival
*Dec 21 21:26:23.620 UTC: %IOS_RESILIENCE-5-CONFIG_RESIL_INACTIVE: Disabled secure config archival [removed
flash:.runcfg-20151221-210831.ar]

*Dec 21 21:27:05.500 UTC: %IOS_RESILIENCE-5-IMAGE_RESIL_ACTIVE: Successfully secured running image
*Dec 21 21:27:11.576 UTC: %IOS_RESILIENCE-5-CONFIG_RESIL_ACTIVE: Successfully secured config archive
[flash:.runcfg-20151221-212711.ar]
```

Blocked Attempts

Any attempt to access a service or run a command that is blocked or denied could be an indication of remote scanning, information gathering or brute-force attempts to gain unauthorized access. The following log messages are examples of actions that were blocked due to disabled services, protocol mismatches, invalid parameters, wrong credentials, or denied due to access lists. It is recommended for all access lists to at least log denied attempts. Unfortunately, logs may not exist for brute-force attempts to guess passwords or SNMP community strings, and may require examining the output of "show snmp" and other commands, or by analyzing network traffic.

```
%RCMD-4-RSHPORTATTEMPT: Attempted to connect to RSHELL from 10.0.0.2

%SSH-4-SSH2_UNEXPECTED: Unexpected message type has arrived. Terminating the connection from 192.168.0.130

%SSH-5-SSH2_SESSION: SSH2 Session request from 10.64.15.82 (tty = 0) using crypto cipher '', hmac '' Failed

%SNMP-3-AUTHFAIL: Authentication failure for SNMP req host 192.168.0.31

%SEC-6-IPACCESSLOGP: list 150 denied tcp 192.168.0.1(48980) -> 192.168.0.34(6666), 1 packet
%SEC-6-IPACCESSLOGNP: list 50 denied 0 192.168.0.33 -> 192.168.0.34, 1 packet
%SEC-6-IPACCESSLOGS: list 25 denied 15 192.168.0.99 3 packets
```

Various types of log messages could indicate that a malicious user was attempting to circumvent network restrictions by injecting packets, spoofing MAC address, changing VTP domains or attempting to connect to existing trunks or channels.

```
%ADJ-3-RESOLVE_REQ: Adj resolve request: Failed to resolve 192.168.0.1 Vlan1

%BGP-3-NOTIFICATION: received from neighbor 10.1.1.2 2/3 (BGP identifier wrong) 4 bytes 0A010101

%C4K_IOSSYS-7-INVALIDVALUE: Platform value type 60 not handled returning a default of 0

%C4K_L2MAN-6-INVALIDSOURCEADDRESSPACKET: Packet received with invalid source MAC address (DE:AD:BE:EF:CA:FE)
on port Gi1/1 in vlan 999

%CDP-4-DUPLEX_MISMATCH: duplex mismatch discovered on FastEthernet 5/1 (not half duplex), with
SEP00225100BEEF Port 1 (half duplex).

%CDP-4-NATIVE_VLAN_MISMATCH: Native VLAN mismatch discovered on GigabitEthernet11/16 (1101), with WS-C2960-
24-TC-S FastEthernet 0/3 (1).

%CHKPT-4-GET_HUGE_BUF: Client 33 buffer request (size=9028) is too large

%CSM_SLB-6-GATEWAYSTATE: Module 3 gateway state changed: SLB-NETMGT: Got different MAC address from Gateway
10.196.13.15 in response to ARP

%DTP-SP-5-DOMAINMISMATCH: Unable to perform trunk negotiation on port Gi0/1 because of VTP domain mismatch.
%SW_VLAN-6-VTP_DOMAIN_NAME_CHG: VTP domain name changed to HACKER.

%EC-5-ERRPROT3: Command rejected: the interface Fa0/1 is already part of a channel

%SW_MATM-4-MACFLAP_NOTIF: Host 0022.15de.ad00 in vlan 99 is flapping between port Fa0/1 and port Fa1/1
```

Downgraded Encryption

When properly configured, SSH version 1 should never be utilized by servers or clients. Any attempts to use version 1 could be an adversary performing a man-in-the-middle attack to downgrade the encryption used during the connection negotiation so that the encrypted data could be easier to crack. Invalid algorithms or unexpected message types could also be an attempt to circumvent security measures provided by the protocol in use. The first log messages below indicate a client successfully utilizing SSH version 1. The last log message indicates a client attempting to connect to the SSH server utilizing only version 1 when the device was configured to only accept version 2 connections.

```
%SSH-5-SSH_SESSION: SSH Session request from 192.168.3.1 (tty = 1) using crypto cipher '3DES' Succeeded
%SEC_LOGIN-5-LOGIN_SUCCESS: Login Success [user: admin] [Source: 192.168.3.1] [localport: 22] at 20:34:24
UTC Wed Dec 23 2015
%SSH-5-SSH_USERAUTH: User 'admin' authentication for SSH Session from 192.168.3.1 (tty = 1) using crypto
cipher '3DES' Succeeded

%SSH-3-NO_MATCH: No matching hostkey algorithm found: client ssh-dss server ssh-rsa
%SSH-4-SSH2_UNEXPECTED_MSG: Unexpected message type has arrived. Terminating the connection from
192.168.2.58

%SSH-5-SSH_CLOSE: SSH Session from 192.168.0.213 (tty = 1) for user '' using crypto cipher '' closed
```

Log messages that indicate a weak protocol has been enabled or certain security features have been disabled for any protocol could also be an indication of unauthorized access.

```
%SSH-5-ENABLED: SSH 1.99 has been enabled

%CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
```

Cisco IOS - Software Modification Detection

File Verification

Cisco IOS systems store the operating system in a single image file. Each image file contains all of the executable code necessary for the network device to operate. It is generally only necessary to store one image file on a device, but some devices may contain more than one image file if the device has recently been upgraded or if the network administrators failed to remove old image files after they were no longer needed.

First it is necessary to determine what files on a Cisco IOS system need to be verified. This should include all the operating system image files stored on the device (**running OS version**, **boot OS versions** and **other OS versions**), but this process can be applied to any file. Image files usually have a ".bin" extension and are large in size (up to hundreds of megabytes). The "show version" command can be used to determine the **file version**, **file name**, and **file path** of the running OS version.

```
Switch>show version
Cisco IOS Software, C3750E Software (C3750E-IPBASEK9-M), Version 12.2(58)SE2, RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2011 by Cisco Systems, Inc.
Compiled Thu 21-Jul-11 01:23 by prod_rel_team

ROM: Bootstrap program is C3750E boot loader
BOOTLDR: C3750E Boot Loader (C3750E-HBOOT-M) Version 12.2(44r)SE3, RELEASE SOFTWARE (fc3)

Switch uptime is 26 weeks, 5 days, 16 hours, 0 minutes
System returned to ROM by power-on
System image file is "flash:/c3750e-ipbasek9-mz.122-58.SE2.bin"
```

For all other files, the "dir all-filesystems" command can be used initially to discover files. Image files can be stored in subdirectories, and there may be multiple image files stored on each device. Subsequent commands using the "dir" command will be necessary to look in any subdirectories, or the "/recursive" parameter can be utilized to recursively examine directories.

```
Switch#dir all-filesystems
Directory of flash:/

    2  -rwx    17866988  May 31 1993 04:15:17 +00:00  c3750e-ipbasek9-mz.122-58.SE2.bin
    6  -rwx        3653  Mar 22 1993 07:27:01 +00:00  config.text
  400  -rwx         556   Mar 1 1993 00:01:03 +00:00  vlan.dat
  402  -rwx        5505  Mar 22 1993 07:27:02 +00:00  private-config.text
  404  -rwx        4120   Mar 1 1993 00:01:09 +00:00  multiple-fs

57671680 bytes total (11082240 bytes free)
```

The "dir" command can be used directly on a specific file. The **file name**, **file path**, **file size**, **file timestamp**, and **file permissions** can be obtained on individual files by using the "dir" command with the entire path and name of a file as a parameter. The **file version** of an image file can usually be assumed by the file name, but this may not be accurate since the file could have been renamed.

```
Switch#dir flash:/c3750e-ipbasek9-mz.122-58.SE2.bin
Directory of flash:/c3750e-ipbasek9-mz.122-58.SE2.bin

    2  -rwx    17866988  May 31 1993 04:15:17 +00:00  c3750e-ipbasek9-mz.122-58.SE2.bin
```

Online Hash

Before obtaining the hashes of a file, a cryptographic **hash type** must be chosen and the same hash type must be used throughout this process for each file. Most Cisco IOS systems should support MD5 hashes, and newer versions of the operating system support as least SHA1. The following examples will show how to obtain the MD5 hashes, but the same process can be repeated for other hashes, if desired. The more hash types utilized for the same file, the higher the confidence.

There are two forms of the online hash on Cisco IOS systems. The first **online hash** is obtained by executing the "verify" command and extracting the "CCO Hash" (**online1**). The other hashes displayed are utilized for self-verification, described below. Newer systems may display SHA1 hashes.

```
Switch#verify sup-bootdisk:/s3223-ipbasek9-mz.122-33.SXI2a.bin
Verifying file integrity of sup-bootdisk:/s3223-ipbasek9-mz.122-33.SXI2a.bin.....................
...................................................................................Done!
Embedded Hash   MD5 : EC9DE37863AE100EC6E32B0DFD8855D9
Computed Hash   MD5 : EC9DE37863AE100EC6E32B0DFD8855D9
CCO Hash        MD5 : 80215AD7A7B6F81439D0B2BB525EA248

Embedded hash verification successful.
```

Unfortunately, some versions of IOS do not display any hashes when using the "verify" command, and only confirm whether or not the verification was successful. For this reason, the online1 hash is not considered a requirement for verifying the integrity of a file. But if the online1 hash is available and does not match the published hash, further investigation is required. Similarly, if the command reports that the verification failed, then the file may have been modified and should also be investigated.

```
Switch#verify bootflash:/cat4500-ipbasek9-mz.122-53.SG.bin
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCC
File system hash verification successful.
```

Additionally, a small number of IOS versions contain a bug and report that the file does not exist when using the "verify" command, resulting in a verification failure. This bug was fixed in later versions of the operating system and can be avoided by properly upgrading those systems.

```
Switch#verify flash:/c3560-advipservicesk9-mz.122-46.SE.bin
File system hash verification failed for file
flash:/c3560-advipservicesk9-mz.122-46.SE.bin(No such file or directory).
```

The second **online hash** is obtained by executing the "verify /md5" command (**online2**), and is the required online hash that should be used in the final comparison. Newer versions may also support other hash types such as SHA1 by using "/sha1" in the command.

```
Switch#verify /md5 sup-bootdisk:/s3223-ipbasek9-mz.122-33.SXI2a.bin
................................................................................................
..........................Done!
verify /md5 (sup-bootdisk:/s3223-ipbasek9-mz.122-33.SXI2a.bin) = 80215ad7a7b6f81439d0b2bb525ea248
```

Offline Hash

The **offline hash** is computed by copying the file from the Cisco IOS system to a trusted system. Various versions of IOS may support the SCP, HTTP, TFTP and FTP protocols. When using clear-text protocols, someone collecting traffic on the network would be able to intercept the credentials and files if they were copied using one of these protocols, so SCP is the preferred protocol to use. All of these protocols require a server to be running on the trusted system. For HTTP, the server must be capable of handling PUT requests. The configurations of these services are system specific and are not covered here.

Files can be copied remotely from Cisco IOS systems by using the "copy" command. When logging into the system over the console or a vty line, the line must be configured to allow outbound connections. If "transport output none" is applied to that specific line, the copy command will fail. When using SCP or FTP, a username and password may be required to remotely connect to the trusted system. A separate temporary account should be used for this purpose, since those credentials must be supplied to the IOS system for it to connect back to the trusted system. The adversary may be able to obtain those credentials if the operating system of that device was modified, so this account and any services should be disabled when they are not being used. If possible, a chroot environment or an invalid shell may be preferred.

When copying a file, the same number of bytes should be transferred that matches the output obtained from the "dir" command (**file size**).

```
Switch#copy flash:/c3560-advipservicesk9-mz.122-46.SE.bin scp://10.16.7.32:22/
Address or name of remote host [10.16.7.32]? 10.16.7.32
Destination username [Switch]? ndi
Destination filename [c3560-advipservicesk9-mz.122-46.SE.bin]? 10.4.7.102_c3560-advipservicesk9-mz.122-
46.SE.bin
Writing 10.4.7.102_c3560-advipservicesk9-mz.122-46.SE.bin
Password:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
10713344 bytes copied in 33.638 secs (318489 bytes/sec)
```

An alternative method of copying files from a Cisco IOS system is enabling the SCP server on the device itself using the "ip scp server enable" global configuration command. This service is not enabled by default, so it must be implicitly enabled. This may be an easier method for obtaining files when connections cannot be established from the device back to the trusted system due to access restrictions implemented on the network.

```
linux$ /usr/bin/scp -p -o StrictHostKeyChecking=no
    admin@10.4.7.102:flash:/c3560-advipservicesk9-mz.122-46.SE.bin \
    10.4.7.102_c3560-advipservicesk9-mz.122-46.SE.bin
```

The final method to obtain a copy of a file is with the "more" command. This can be utilized when the file cannot be transferred via an external connection, such as over the console port, or when inbound and outbound connections are disallowed. Unfortunately, this method will be extremely slow, especially over the console port, and will require converting the entire output from hexadecimal back to binary.

```
Router#more flash:/c2801-advipservicesk9-mz.124-21a.bin
00000000:  7F454C46 01020100 00000000 00000000    .ELF .... .... ....
00000010:  00020086 00000001 8000F000 00000034    .... .... ..p. ...4
00000020:  00000054 20002001 00340020 00010028    ...T  . . .4.  ...(
00000030:  00070000 00000001 0000016C 8000F000    .... .... ...l ..p.
00000040:  8000F000 01D6B31C 01D93BBC 00000007    ..p. .V3. .Y;< ....
```

Once the file is obtained from the device, the **offline hash** can be computed by executing the appropriate command to compute the cryptographic hash of the file, such as "md5sum".

```
linux$ md5sum 10.4.7.102_c3560-advipservicesk9-mz.122-46.SE.bin
47807a7e5e5eed4ba84e839b2f8c7979  10.4.7.102_c3560-advipservicesk9-mz.122-46.SE.bin
```

Published Hash (Known Good)

The **published hash** can be obtained from Cisco's website at www.cisco.com. Individual hashes can be obtained by browsing under the Support and Download links. Additionally, a Cisco Security Advisory was released in 2008 titled *Rootkits on Cisco IOS Devices*[8], which provides directions for performing a similar file verification process. This page also provides a ZIP file that contains CSV files with a large number of hashes of the IOS image files released up to a certain date.

http://www.cisco.com/c/dam/assets/about/security/resources/ioshashes.zip

The CSV files do not contain the **published file path**, **published file version**, **published file size** or the **published release data**. The date listed in the CSV files is when the image file was added to the list, which may not necessarily be the published release date. All of this other information can be obtained from Cisco's website, but each image file must be searched for individually.

Hashes can also be obtained from the Bulk Hash File Download Center[9], which provides a TAR file that contains a CSV file with names for image files and other software files published by Cisco, and includes both MD5 and SHA512 hashes, **published release date** and **published file size** for each file.

http://www.cisco.com/c/en/us/support/web/tools/bulk-hash/index.html

Even though the TAR file contains a significantly smaller number of hashes than the ZIP file, it does contain hashes that are not included in the ZIP file. Unfortunately, both of the above sources may not always be updated on a regular basis, and may not contain information for all published image files. It still may be necessary to search for specific images on Cisco's website to obtain the published hash.

It is critical to ensure the correct published hash is obtained. A file can be renamed to anything, so it may not match exactly. It may be necessary to obtain the version information from the output of "show version" to ensure the correct hash is used. If a hash does not match, it should also be compared with other known hashes to see if the file was inadvertently renamed to another valid file name.

It is possible that a single file name may have multiple published hashes associated with it. Cisco may have different versions of the same file name based on the model, or maybe there were minor updates to a file before it was actually released. Regardless, it may be necessary to utilize more than one published hash when verifying the integrity of a file, but obviously only one of the published hashes should match if the file has not been modified.

Hash Comparison

After the online, offline, and published hashes have been obtained, they can be compared for equality by following the guidance previously provided. Any inconsistencies should be further investigated to determine the cause, which may be the result of padding, corruption or a malicious file modification.

Self-Verification

The self-verification feature of Cisco IOS generates a **computed hash** on specific portions of an image file and compares the result with an **embedded hash** stored in the image file itself. This feature can be utilized with the "verify" command, or can be automatically utilized when the "copy" or "reload" commands are used. This feature can be enabled with the "file verify auto" global configuration command.

```
Switch#verify sup-bootdisk:/s3223-ipbasek9-mz.122-33.SXI2a.bin
Verifying file integrity of sup-bootdisk:/s3223-ipbasek9-mz.122-33.SXI2a.bin....................
...................................................................................Done!
Embedded Hash   MD5 : EC9DE37863AE100EC6E32B0DFD8855D9
Computed Hash   MD5 : EC9DE37863AE100EC6E32B0DFD8855D9
CCO Hash        MD5 : 80215AD7A7B6F81439D0B2BB525EA248

Embedded hash verification successful.
```

As previously stated, some versions of IOS do not display the hashes when using the "verify" command, and only confirms whether or not the verification was successful.

```
Switch#verify bootflash:/cat4500-ipbasek9-mz.122-53.SG.bin
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCC
File system hash verification successful.
```

Also stated previously, a small number of IOS versions contain a bug and report that the file does not exist when using the "verify" command, resulting in a verification failure. This bug was fixed in later versions of operating system and can be avoided by properly upgrading those systems.

```
Switch#verify flash:/c3560-advipservicesk9-mz.122-46.SE.bin
File system hash verification failed for file
flash:/c3560-advipservicesk9-mz.122-46.SE.bin(No such file or directory).
```

Unfortunately, the result of any self-verification feature cannot necessarily be trusted. The embedded hash is not published on the vendor's website, and an adversary capable of modifying the image file is also capable of modifying the hash embedded in the file to match the expected result. Regardless, if the "verify" command produces a failed result or the embedded and computed hashes are not equal, then the file has been modified from the original. It does not necessarily mean that the file has been modified maliciously, but it is an indicator that the file will need to be investigated. The file verification process described above utilizing the online and offline hashes should be followed regardless of the result of self-verification.

In addition to the self-verification feature described above, the undocumented Cisco IOS command "show sum" can be utilized to display the checksum for the current running image. This can be considered as another method to perform self-verification. Again, it does not provide a high level of confidence but can also be a quick indication that something may have been modified if it returns a negative result, or can be included in the process to provide a higher level of confidence.

```
Switch#show sum
New checksum of 0xED9EC8B6 matched original checksum
```

Furthermore, some platforms support digitally signed images[4]. The authenticity and integrity of a binary file can be verified with the "show software authenticity file" command or on the currently booted image with "show software authenticity running". Currently stored keys[10] can be obtained with the command "show software authenticity keys".

```
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
```

Memory Verification

The memory on a Cisco IOS system can be verified similar to how files are verified. The entire contents of memory can be accessed via system:/memory/main. Unfortunately, the contents of memory are always changing, so computing a cryptographic hash of memory is useless. Only the portion of memory that contains executable code is necessary for verifying the integrity of the software running on the device. Fortunately, this area of memory can be accessed directly with system:/memory/text.

```
Router#dir system:/memory/
Directory of system:/memory/

  150  -r--     3956768                  <no date>  bss
  149  -r--    19917824                  <no date>  data
  151  -r--   296925856                  <no date>  heap
  146  -r--    41943040                  <no date>  iomem
  147  -r--   360710144                  <no date>  main
  152  -r--   360710144                  <no date>  main_k0
  153  -r--   360710144                  <no date>  main_k1
  148  -r--    39784448                  <no date>  text
```

```
Switch#dir system:/memory/
Directory of system:/memory/

   11  -r--    10888608                  <no date>  bss
    6  -r--   134217696                  <no date>  coredump
    9  -r--    10255844                  <no date>  data
    8  -r--     1048576                  <no date>  dltext
   12  -r--    34430072                  <no date>  heap
   13  -r--    12582912                  <no date>  iomem
    5  -r--   134217728                  <no date>  main
   10  -r--      262144                  <no date>  reclaimed_heap
    7  -r--    31190300                  <no date>  text
```

The text portion of memory referenced by system:/memory/text contains the executable code used by the operating system. If this area of memory is consistent every time the devices boots and does not change, a cryptographic hash can be computed on this area of memory just like a static file.

Online Hash

The **online hash** of the text portion of memory can be computed the same as other files with the verify command specifying the type of cryptographic hash to compute.

```
Router#verify /md5 system:/memory/text
............................................................................................................
.................................Done!
verify /md5 (system:/memory/text) = 0f8ade529955186186cb7e6d62e12fcf
```

```
Switch#verify /md5 system:/memory/text
............................................................................................................
...............Done!
verify /md5 (system:/memory/text) = 26d455e40cc795da7bbea174407055ea
```

Offline Hash

The **offline hash** of the text portion of memory from system:/memory/text can be computed the same as other files by copying the memory off of the device to a trusted system with the copy command.

```
Router#copy system:/memory/text scp://cisco@192.168.0.129/
Address or name of remote host [192.168.0.129]?
Destination username [cisco]?
Destination filename [text]? Router-text
Writing Router-text
Password:
 Sink: C0644 39784448 Router-text
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
39784448 bytes copied in 94.980 secs (418872 bytes/sec)
```

```
Switch#copy system:/memory/text scp://cisco@192.168.0.129/
Address or name of remote host [192.168.0.129]?
Destination username [cisco]?
Destination filename [text]? Switch-text
Writing Switch-text
Password:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!
31190300 bytes copied in 72.612 secs (429547 bytes/sec)
```

The entire contents of memory can also be obtained by using the copy command with system:/memory/main. ****WARNING**** **-** This command should only be used on **routers**. Do NOT attempt to use the verify or copy commands on system:/memory/main on **switches**. These commands will always cause a switch to crash and reload, regardless of model or operating system version.

```
Router#copy system:/memory/main scp://cisco@192.168.0.129/
Address or name of remote host [192.168.0.129]?
Destination username [cisco]?
Destination filename [main]? Router-main
Writing Router-main
Password:
 Sink: C0644 360710144 Router-main
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!

360710144 bytes copied in 831.780 secs (433661 bytes/sec)
```

Alternatively, the entire contents of memory can be copied from both switches and routers with the "write core" command. Generally only a limited number of methods can be utilized to copy files, such as the Trivial File Transfer Protocol (TFTP) or File Transfer Protocol (FTP)[11]. When using the default of TFTP, the listening service may need to be configured to accept the creation of new files, as the device may choose a file name based on the current time, such as "Switch-corecoredump_20151228-192714", which cannot be determined directly from the output provided below.

```
Switch#write core 192.168.0.129
Base name of core files to write [Switch-core]?
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!
```

If the entire memory is obtained by copying system:/memory/main or with "write core", the text portion can be extracted if the correct offsets are known[4]. Most of this information can be retrieved from the "show region" command. The start position of main can be subtracted from the start position of text to determine the byte position in main where to begin the extraction. The size of the text portion is listed in the output, or it can be calculated by an inclusive difference of the start and end positions. When extracted from system:/memory/main, the text portion of memory should be exactly the same as system:/memory/text.

If the memory is obtained with the "write core" command, it may be necessary to map virtual memory addresses to physical memory addresses to obtain the actual location of the main and text portions of memory. This can be achieved by looking at the translation lookaside buffer (tlb) with the "show platform", "show platform tlb" or "show platform hardware tlb" commands[12]. If these commands are not available, the start of the core can be assumed to also be the start of the main portion of memory.

Also, the text portion of memory contains executable code that should not be changed, so it is reasonable for the operating system to protect that area of memory by marking it as read-only (R/O) instead of read-write (R/W). Unfortunately, this is not the case for all systems, as shown below, so this cannot be used as an indicator of a malicious software modification.

In the Router example below, the text portion of memory resides at 0x6000F000, and the main memory resides at 0x60000000. Thus the text portion of memory resides at position 0x0000F000 (61440) in the system:/memory/main file that was obtained, and is 39784448 bytes long. According to the TLB, virtual memory with an address of 0x60000000 is physically located at 0x00000000, so the calculated offset should not be affected if the memory was retrieved with the "write core" command.

```
Router#show region
Region Manager:

      Start         End     Size(b)  Class  Media  Name
 0x15800000  0x17FFFFFF    41943040  Iomem  R/W    iomem:(iomem)
 0x60000000  0x757FFFFF   360710144  Local  R/W    main
 0x6000F000  0x625FFFFF    39784448  IText  R/O    main:text
 0x6260F940  0x6390E53F    19917824  IData  R/W    main:data
 0x6390E540  0x63CD455F     3956768  IBss   R/W    main:bss
 0x63CD4560  0x757FFFFF   296925856  Local  R/W    main:heap
 0x80000000  0x957FFFFF   360710144  Local  R/W    main:(main_k0)
 0xA0000000  0xB57FFFFF   360710144  Local  R/W    main:(main_k1)
 0xF5800000  0xF7FFFFFF    41943040  Iomem  R/W    iomem
```

```
Router#show platform

TLB entries :
Size  Virt Address range      Phy Address range       Attributes
 16M  0x52000000:0x53FFFFFF   0x72000000:0x73FFFFFF   CacheMode=2, RW, Valid
 16M  0x60000000:0x61FFFFFF   0x00000000:0x01FFFFFF   CacheMode=3, RO, Valid
  1M  0x62000000:0x621FFFFF   0x02000000:0x021FFFFF   CacheMode=3, RO, Valid
```

In the Switch example below, the text portion of memory resides at 0x01000000, and the main memory (coredump) resides at 0x00000020. Thus the text portion of memory resides at position 0x00FFFFE0 (16777184) in the system:/memory/main file that was obtained, and is 31190300 bytes long. Note that the size of the text potion of memory is not divisible by 512 bytes in this case. Note that the "show platform" commands were not valid on this system.

```
Switch#show region
Region Manager:

      Start         End      Size(b)  Class  Media  Name
 0x00000000  0x07FFFFFF   134217728  Local  R/W    main
 0x00000020  0x07FFFFFF   134217696  Local  R/W    main:coredump
 0x01000000  0x02DBED1B    31190300  IText  R/W    coredump:text
 0x02E00000  0x02EFFFFF     1048576  IText  R/W    coredump:dltext
 0x02F00000  0x038C7DE3    10255844  IData  R/W    coredump:data
 0x03599654  0x035D9653      262144  Local  R/W    data:reclaimed_heap
 0x038C7DE4  0x0432A383    10888608  IBss   R/W    coredump:bss
 0x0432A388  0x063FFFFF    34430072  Local  R/W    coredump:heap
 0x06400000  0x06FFFFFF    12582912  Iomem  R/W    coredump:iomem
```

Once the offsets are calculated, the text portion of memory can be extracted from main. A block size of 1 is chosen below for simplicity since the number of bytes to skip and the size of the text portion of memory may not be divisible by 512 bytes. Unfortunately, using "dd" with a block size of 1 is significantly slower, but will provide the most accurate results based on the skip and count values because it reduces the probability of error in calculating the positions and size.

```
Linux$ dd if=Router-main bs=1 skip=61440 count=39784448 of=Router-text_extracted
39784448+0 records in
39784448+0 records out
39784448 bytes (40 MB) copied, 72.977 s, 545 kB/s
```

```
linux$ dd if=Switch-corecoredump_20151228-192714 bs=1 skip=16777216 count=31190300 of=Switch-text_extracted
31190300+0 records in
31190300+0 records out
31190300 bytes (31 MB) copied, 57.0028 s, 547 kB/s
```

Once an offline copy of the text portion of memory is obtained either by copying or extracting, the **offline hash** can be calculated the same as other files. In both of these cases, the hashes match the corresponding online hashes listed above for system:/memory/text.

```
linux$ md5sum Router-text*
0f8ade529955186186cb7e6d62e12fcf  Router-text
0f8ade529955186186cb7e6d62e12fcf  Router-text_extracted
```

```
linux$ md5sum Switch-text*
26d455e40cc795da7bbea174407055ea  Switch-text
26d455e40cc795da7bbea174407055ea  Switch-text_extracted
```

Published Hash (Known Good)

Unfortunately, there is no direct method of obtaining a **published hash** for the text portion of memory from the vendor, or any of the other **published** fields. The only reasonable method for obtaining a known good hash is to boot an identical system with a known good operating system image file and manually compute the hash and retain it for future reference. This is assuming the hardware or firmware has not been tampered with.

Hash Comparison

To verify the integrity of memory without a published hash, the obtained hashes can be compared with other similar systems running the same version of the operating system, if available. Unfortunately, this does not necessarily provide a high level of confidence. If all the systems running the same operating system version were compromised, it would be difficult to detect a difference in any of them. Assuming that the probability is low for all systems with the same operating system version to be compromised, this is a reasonable method of verifying the cryptographic hashes of the executable portion of memory.

A more complicated method to verify the integrity of memory is to compare the contents of memory with a known good image file. This method provides a significantly higher level of confidence than comparing hashes to like systems. If file verification was performed on the image file that was supposedly running in memory, and the verification was successful, the same file that was copied for obtaining the offline hash can theoretically be used as a reference to determine what the text portion of memory should contain. Alternatively, a known good image file can be obtained from Cisco's website if a service contract is available for the given system type. The technical process for comparing an operating system image file with the contents of memory is potentially available from other sources[13] [14] [15]. The Network Appliance Forensic Toolkit (NAFT)[16] and Cisco Incident Response (CIR)[17] are two implementations that may be capable of comparing a known good image with the contents of memory from some systems.

Some Cisco IOS systems implement address space layout randomization (ASLR) which would result in the location and contents of system:/memory/text to be different every time the device boots. Obviously a cryptographic hash would not be sufficient for verifying the integrity of the contents of memory when ASLR is implemented since the contents are expected to be different. Comparing the contents of memory with a known good image file would be necessary to perform memory verification since all of the offsets within memory would be different for every instance, assuming that the process for determining the offsets to compensate for ASLR can be achieved. Unfortunately, in the Router example provided above, the system is known to implement ASLR. So the hash listed above will always be different for systems running the same operating system version and cannot be retained and referenced as a known good.

Even though memory verification is focused on verifying system:/memory/text, it may be necessary to also verify other areas of memory, such as system:/memory/data. Tcl and other rootkit modifications may exist in the data section of memory[18] [19], and a modification to system:/memory/text may not be necessary to invoke those modifications. Additional research is still required to determine if this area of memory can be adequately verified.

Self-Verification

A **self-verification** feature does not exist for the contents of memory because it is not an image file and it does not contain an **embedded hash**. The "verify" command cannot be used without a hash type parameter (such as /md5 or /sha1) on any file from the system:/ file system. Regardless, some additional verification steps can be performed to ensure there are no processes attempting to execute code outside of the text portion of memory.

This can be achieved by verifying that the program counter (PC) and return address (RA) of each process are within the text portion of memory[20]. The output of "show region" can be used to determine where in memory the text portion resides, and the output of "show processes" and "show stacks" can be utilized to determine the PC and RA of each process. The "show stacks" command must be repeated for every process listed in "show processes" and can be a tedious process because of the large number of processes running on a Cisco IOS system. In a previous Switch example, the text portion of memory was located between 0x01000000 and 0x02DBED1B. All of the PC and RA values are within this range for the process with a PID of 4.

```
Switch#show processes
CPU utilization for five seconds: 5%/0%; one minute: 6%; five minutes: 6%
 PID QTy       PC Runtime(ms)   Invoked   uSecs    Stacks   TTY Process
   1 Cwe   2ACD17C           0        22       0  5460/6000    0 Chunk Manager
   2 Csp   1BB5AD0          17      2241       7  2588/3000    0 Load Meter
   3 Mwe   204649C           0        97       0  5620/6000    0 DHCPD Timer
   4 Lst   2ADBAE8      110283      5080   21709  5740/6000    0 Check heaps
   5 Cwe   2AE398C           0         2       0  5524/6000    0 Pool Manager

Switch#show stacks 4
Process 4:  Check heaps
  Stack segment 0x4364684 - 0x4365DF4
  FP: 0x4365DC8, RA: 0x2AE8340
  FP: 0x4365DE0, RA: 0x2ADBAEC
  FP: 0x4365DE8, RA: 0x1BBE468
  FP: 0x0, RA: 0x1BB4EE0
```

Firmware Verification

Unfortunately, there are no known traditional methods that can be utilized to verify the integrity of the firmware stored on a Cisco IOS system. Even though the ROMMON can be upgraded on specific models, there are no direct methods to hash or extract the contents of the ROMMON from an operational system. It may be possible to verify the contents of the ROMMON in the ROM monitor mode before the IOS operating system image file is booted, but this would require physical access to the device and the system would need to be removed from the network.

Regardless, it is not common for the ROMMON to be upgraded on Cisco IOS systems, unless there is a specific boot problem that needs to be addressed. The ROMMON has been upgraded on a router if there are multiple **firmware versions** stored on the device. And even though there are legitimate reasons to upgrade the ROMMON of a system, this could potentially be an indicator that an unauthorized modification has been made to the system. Fortunately, if the ROMMON has been modified maliciously, the intent of those modifications would likely be to inject code into the operating system after the device boots, resulting in a memory only implant which could theoretically be detected using the above memory verification.

Rootkit Detection

The SANS Institute previously released a white paper[21] detailing an attack to hide portions of the configuration from unsuspecting administrators by running Tool Command Language (Tcl) scripts on supported Cisco IOS devices, and more recent research has shown how an adversary can develop their own Tcl commands[18]. **Rootkit detection** can be performed by comparing the same information from multiple sources and looking for discrepancies. For Cisco IOS systems, this process can be performed on the configuration, as well as other attributes of the system, such as interface status, file listings, or established network connections. The process described below is only for comparing the configuration obtained through multiple methods. If a discrepancy is discovered here, the operating system software should be verified both on disk and in memory, and the existence of any suspicious files or scripts should be investigated.

Please note that this is NOT comparing the running-config with the startup-config, which would only detect when a change in the running-config was not permanently saved in the startup-config. This scenario could be a potential vulnerability because a critical configuration change would no longer be present if the device lost power or rebooted, such as sticky MAC addresses utilized by port-security, but such an issue would not be the result of a rootkit.

Malicious behavior that hides portions of the configuration can be detected by obtaining each configuration through multiple methods and comparing the output to look for differences. Any discrepancies between the different outputs can be an indication that something is attempting to hide that information. Obtaining the configurations by copying the files from the network device to a trusted system can provide a higher level of confidence since it would be difficult for an adversary to modify the copy functions used by the operating systems, especially through Tcl scripts such as described above. Regardless, the more methods used to obtain the same information, the higher the confidence that the provided information was unmodified by an adversary. The following table lists the various commands that can be executed to obtain both the running and startup configurations from a Cisco IOS device. All of these commands may not be supported on all systems.

Cisco IOS Rootkit Detection Commands	
running configuration	**startup configuration**
`show running-config`	`show startup-config`
`write term`	`show config`
`more system:/running-config`	`more nvram:/startup-config`
`copy system:/running-config ...`	`copy nvram:/startup-config ...`
`show platform configuration running`	`show platform configuration startup`

After obtaining the configurations using the above commands, it may be necessary to first strip out unnecessary information that should not be included in the comparison. Some common items to ignore could be carriage return (CR) and line feed (LF) differences, comments, blank lines, control characters and configuration headers. Even though the existence of an extra comment could be considered a discrepancy, the initial comparison should exclude these items since they are not processed by the system. Some other pieces of information to consider are the NTP clock period, which may change periodically in the running configuration on some systems. Several Perl compatible regular expressions listed below are useful for removing several of these features.

```
# remove headers
s/^Using \d+ out of \d+ bytes(?:, uncompressed size = \d+ bytes)?$//m;
s/^Building configuration\.\.\.$//m;
s/^Current configuration[ ]?:[ ]*(?:\d+ bytes)?[ ]*$//m;
s/^Uncompressed configuration from \d+ bytes to \d+ bytes$//m;

# remove comments
s/^!([^\r\n]+)*\r?\n//gms;

# reduce false positives
s/^ntp clock-period \d+\r?\n//gms;

# remove blank lines (should only affect the banner)
s/^\r?\n//gms;

# fix control characters
s/\x5e(.)/chr(ord($1)-ord('A')+1)/ge;

# remove CRs
s/\r\n/\n/g;
```

After applying these regular expressions and any other fixes, the data obtained from each of the corresponding commands above can be compared for differences. A basic example is shown below that highlights a difference discovered between two of the outputs, indicating that something was hiding some of the information displayed in "show running-config".

```
Switch>show running-config
hostname Switch
boot-start-marker
boot-end-marker
enable secret 5 $1$yeQk$IxvUdPDgFUtrHTS.1R4ys1
username cisco password 7 030752180500
aaa new-model
```

```
Switch>more system:/running-config
hostname Switch
boot-start-marker
boot-end-marker
enable secret 5 $1$yeQk$IxvUdPDgFUtrHTS.1R4ys1
username cisco password 7 030752180500
username r00t password 7 1405425B18
aaa new-model
```

Cisco IOS - Hardware Modification Detection

Unique Identifiers

The serial number is a hard-coded value assigned to a single piece of equipment. A single system may contain multiple pieces of equipment. Each individual piece of equipment should contain its own unique serial number. Even though it does not guarantee the equipment is legitimate, a basic verification check can be performed on the serial number assigned to each piece of equipment to ensure that it conforms to the expected format.

A serial number from a modern piece of equipment produced by Cisco Systems, Inc should contain exactly 11 characters in the form of **LLLYYWWSSSS**. Older serial numbers consisted of only numbers.

- **LLL** - Location, depends on the manufacturer, should only contain letters.
- **YY** - The number of years the manufacturer has been a Cisco supplier, should only contain numbers.
- **WW** - The week number the product was built, should only contain numbers and not be greater than 53.
- **SSSS** - A unique alphanumeric identifier.

The following logic can be utilized to detect an invalid serial number.

- If the serial number is not exactly 11 characters long, then the serial number is invalid.
- If **LLL** contains any characters besides uppercase letters, then the serial number is invalid.
- If **YY** contains any characters besides numbers, then the serial number is invalid.
- If **WW** contains any characters besides numbers, or **WW** is greater than 53, then the serial number is invalid.
- If **SSS** contains any characters besides numbers and uppercase characters, then the serial number is invalid.

In addition to verifying that all serial numbers conform to the specification, duplicate serial numbers can be detected across a network by combining every **serial number** assigned to every piece of equipment on a single system with the **unique attribute** obtained for that system. The unique attribute should be independent from any of the hardware, such as a cryptographic hash of the stored configuration (nvram:/startup-config). If a serial number corresponds to more than one unique attribute, then that serial number is a duplicate because it is assigned to more than one physically unique system.

Similar to serial numbers, the media access control (MAC) address of a network interface card can be combined with the **unique attribute** of a given system. Any MAC addresses that is associated with more than one unique attribute could be indication of a duplicate MAC address. This could indicate that one of the network interface cards, or even one of the entire systems, could be counterfeit. It may be necessary to exclude the MAC addresses of virtual interfaces (such as VLANs), even though these MAC addresses generally correspond with a physical interface.

Operating Statistics

Unfortunately, additional research may be required to understand how a hardware modification can be detected on a Cisco IOS system by looking at **operating statistics**. There does not appear to be sufficient information available to determine when a specific value is out of range, and whether or not this can be adequately utilized to detect a hardware modification. Regardless, this information can still be easily obtained for future reference and comparison.

Some **operating statistics** can be obtained with the "show environment all" or "show env all" commands, depending on the type of system. Unfortunately, the output of these commands may not be very detailed and can vary greatly depending on the platform.

```
Router>show environment all

 Fan  1 Running sucessfully

 Fan  2 Running sucessfully

 ILP  Power Supply -Absent
```

```
Switch>show env all
FAN is OK
TEMPERATURE is OK
SW  PID                 Serial#     Status           Sys Pwr  PoE Pwr  Watts
--  ------------------  ----------  ---------------  -------  -------  -----
 1  Built-in                                         Good

SW  Status          RPS Name          RPS Serial#  RPS Port#
--  -------------   ----------------  -----------  ---------
1   Not Present     <>
```

Some diagnostic tests can be performed on switches with "diagnostic start switch". Unfortunately, some of the tests may require that the device be automatically rebooted after the test is complete. After it reloads, "show diagnostic switch all" can be used to obtain the results. The output of "show diagnostic content switch all" can also be useful.

```
Switch#diagnostic start switch 1 test all
Diagnostic[Switch 1]: Running test(s) 2-6 will cause the switch under test to reload after completion of the
test list.
Diagnostic[Switch 1]: Running test(s) 2-6 may disrupt normal system operation
Do you want to continue? [no]: yes
```

```
Switch>show diagnostic switch all

Switch 1:   SerialNo : FHK1016Y05U

  Overall diagnostic result: PASS

  Test results: (. = Pass, F = Fail, U = Untested)

    1) TestPortAsicStackPortLoopback ---> .
    2) TestPortAsicLoopback ------------> .
    3) TestPortAsicCam -----------------> .
    4) TestPortAsicRingLoopback --------> .
    5) TestMicRingLoopback -------------> .
    6) TestPortAsicMem -----------------> .
```

Cisco IOS - Forensic Evidence

In addition to all of the other commands previously described, several pieces of critical information should be obtained from a compromised network device. The following commands can be utilized to obtain some of this additional critical information. For simplicity, sample output has not been provided for the commands listed below. Additional information about each of these Cisco IOS commands should be available by searching on cisco.com.

The current system clock can be obtained with "show clock detail". This command will display the same information as "show clock" but will also provide the time source.

Processes, memory, and CPU usage can be obtained with the following commands. Please note that the "show memory" command can return a significant amount of information.

show process memory show process cpu	show process cpu history show memory statistics	show memory

Logged in users, Authentication, Authorization and Accounting (AAA) server status, NTP status and other user and login information can be obtained with the following commands.

show aaa servers show aaa method-lists all show aaa sessions show aaa user all show aaa local user lockout show accounting	show whoami show users show users all show radius statistics show tacacs show priv	show privilege show login show ntp status show ntp associations show ntp authentication-status

Status and usage statistics from network interfaces and port security can be obtained with the following commands.

show interfaces show ip interface show ip interface brief show ipv6 interface	show interfaces trunk show interfaces switchport show port-security	show ip traffic show ipv6 traffic show frame-relay map

Other interface status information can be obtained with the following commands, such as Virtual Local Area Network (VLAN) usage, VLAN Trunking Protocol (VTP), Dynamic Trunking Protocol (DTP) and monitor sessions.

show vlan show vlans show vlan-switch show vtp show vtp counters	show vtp status show vtp password show dtp show dtp interface	show spanning-tree show monitor show monitor session 1 show monitor session 2

Status of network connections and network services can be obtained with the following commands.

```
show udp
show tcp
show tcp brief
show tcp brief all
show ip sockets
show protocols
show hosts
show control-plane host open-ports
show crypto key mypubkey rsa
show crypto ca certificates
show crypto pki certificates
show crypto map
show snmp
show snmp group
show snmp user
show ssh
show ip ssh
show ip http server all
show subsys name *
show subsys name http
show ipv6 tunnel
show ipv6 features global
show ipv6 general-prefix
show ipv6 local pool
show ipv6 mtu
show ipv6 prefix-list
```

Neighboring systems can be obtained with the following commands.

```
show cdp
show cdp neighbor
show cdp neighbor detail
show arp
show ip arp
show mac-address-table
show mac address-table
show ipv6 neighbors
show switch neighbors
show ip dhcp database
show ip dhcp pool
show ip dhcp server statistics
show ipv6 dhcp
show ipv6 dhcp database
show ipv6 dhcp interface
show ipv6 dhcp pool
```

Routing information can be obtained with the following commands, including specific routing protocols such as OSPF, EIGRP, BGP ISIS and RIP. Older systems utilize "show ip routing" instead of "show ip route", and thus "show ip rout" should cover either command.

```
show ip rout
show ipv6 route
show ipv6 routers
show ipv6 static
show ip protocols
show ipv6 protocols
show ip cef
show ipv6 cef
show standby
show ip vrf
show ip mroute
show ipv6 mfib
show ipv6 mroute
show ipv6 pim interface
show ip nat translations
show ip nat translations verbose
show ip flow accel
show ip flow export
show ip igmp groups
show mpls traffic-eng tunnels
show ip ospf interface
show ip ospf neighbor
show ip ospf border-routers
show ip ospf database
show ip ospf traffic
show ipv6 ospf interface
show ipv6 ospf neighbor
show ipv6 ospf border-routers
show ipv6 ospf database
show ipv6 ospf traffic
show ip eigrp interfaces
show ip eigrp neighbors
show ip eigrp topology
show ip eigrp traffic
show ipv6 eigrp interfaces
show ipv6 eigrp neighbors
show ipv6 eigrp topology
show ipv6 eigrp traffic
show bgp all
show bgp all neighbors
show bgp all paths
show bgp all summary
show isis database
show isis neighbors
show isis route
show isis topology
show ip rip database
show ipv6 rip
show ipv6 rip database
show ipv6 rip next-hops
```

Access list and traffic statistics can be obtained with the following commands, including how many packets have matched individual rules. If a particular rule does not list any matches, it was never matched.

```
show access-lists
show ip access-lists
show ipv6 access-list
show mac access-group
show ip audit all
show ip cache flow
```

The "show debugging" command can be utilized to see any currently enabled debugging options.

Finally, the "show tech-support" command can be utilized to obtain a significant amount of information about a system which can be useful during a forensic investigation. Most of this information may duplicate what was obtained from other commands, but the output is generally sanitized (such as the removal of passwords, hashes and keys) so it can be provided to a Cisco representative during a support call, if necessary.

Cisco IOS - Command Summary

All of the commands shown above have been summarized here for simplicity. Some of these commands require additional parameters (in *CAPITAL ITALICS*) that can only be determined after examining the output of other commands utilized on the specific system under examination, such as an image file name or a process ID (PID). These same commands may also need to be repeated if there are multiple input parameters, such as the multiple lines available on a system.

```
Show version
show hardware
show inventory raw
show inventory
show diag
show idprom all
show module
show diagbus
show rom-monitor
verify /md5 nvram:/startup-config
show running-config
show running-config all
show interfaces
show ip interface
show ip interface brief
show protocols
show ipv6 interface
show startup-config
show boot
show bootvar
show flash
dir .
dir all-filesystems
dir /recursive /all FILESYSTEM
dir FILE
show bootflash
show slot0
show file systems
show environment all
show env all
show diagnostic switch all
show diagnostic content switch all
show logging
show ntp status
show ntp associations
show arp
show ip arp
show mac address-table
show mac-address-table
show line con 0
show line
show line LINE
show terminal
show usb port
show usb tree
show usb device
show usb controllers
show flash: filesys
more CRASHINFO_FILE
show context
show history all
show history
dir IMAGE_FILE
show secure bootset
verify IMAGE_FILE
verify /md5 IMAGE_FILE
copy IMAGE_FILE REMOTE_URL
show sum
show software authenticity file FILE
show software authenticity running
show software authenticity keys
dir system:/memory/
verify /md5 system:/memory/text
copy /md5 system:/memory/text
write core REMOTE_IP
show region
show platform
```

```
show vlans
show vlan-switch
show vtp
show vtp counters
show vtp status
show vtp password
show dtp
show dtp interface
show udp
show tcp
show tcp brief
show tcp brief all
show ip sockets
show hosts
show control-plane host open-ports
show crypto key mypubkey rsa
show crypto ca certificates
show crypto pki certificates
show crypto map
show snmp
show snmp group
show snmp user
show ssh
show ip ssh
show ip http server all
show subsys name *
show subsys name http
show ipv6 tunnel
show ipv6 features global
show ipv6 general-prefix
show ipv6 local pool
show ipv6 mtu
show ipv6 prefix-list
show spanning-tree
show monitor
show monitor session 1
show monitor session 2
show cdp
show cdp neighbor
show cdp neighbor detail
show ipv6 neighbors
show switch neighbors
show ip dhcp database
show ip dhcp pool
show ip dhcp server statistics
show ipv6 dhcp
show ipv6 dhcp database
show ipv6 dhcp interface
show ipv6 dhcp pool
show ip rout
show ipv6 route
show ipv6 routers
show ipv6 static
show ip protocols
show ipv6 protocols
show ip cef
show ipv6 cef
show standby
show ip vrf
show ip mroute
show ipv6 mfib
show ipv6 mroute
show ipv6 pim interface
show ip nat translations
show ip nat translations verbose
show ip flow accel
show ip flow export
```

show platform tlb	show ip igmp groups
show platform hardware tlb	show mpls traffic-eng tunnels
show processes	show ip ospf interface
show stacks *PID*	show ip ospf neighbor
write term	show ip ospf border-routers
more system:/running-config	show ip ospf database
copy system:/running-config *REMOTE_URL*	show ip ospf traffic
show platform configuration running	show ipv6 ospf interface
show config	show ipv6 ospf neighbor
more nvram:/startup-config	show ipv6 ospf border-routers
copy nvram:/startup-config *REMOTE_URL*	show ipv6 ospf database
show platform configuration startup	show ipv6 ospf traffic
show clock	show ip eigrp interfaces
show clock detail	show ip eigrp neighbors
show process memory	show ip eigrp topology
show process cpu	show ip eigrp traffic
show process cpu history	show ipv6 eigrp interfaces
show memory statistics	show ipv6 eigrp neighbors
show memory	show ipv6 eigrp topology
show aaa servers	show ipv6 eigrp traffic
show aaa method-lists all	show bgp all
show aaa sessions	show bgp all neighbors
show aaa user all	show bgp all paths
show aaa local user lockout	show bgp all summary
show accounting	show isis database
show whoami	show isis neighbors
show users	show isis route
show users all	show isis topology
show radius statistics	show ip rip database
show tacacs	show ipv6 rip
show priv	show ipv6 rip database
show privilege	show ipv6 rip next-hops
show login	show access-lists
show ntp authentication-status	show ip access-lists
show interfaces trunk	show ipv6 access-list
show interfaces switchport	show mac access-group
show port-security	show ip audit all
show ip traffic	show ip cache flow
show ipv6 traffic	show debugging
show frame-relay map	show tech-support
show vlan	

References

[1] Cisco IOS Configuration Fundamentals Command Reference - show protocols through showmon [No Date]
http://www.cisco.com/c/en/us/td/docs/ios/fundamentals/command/reference/cf_book/cf_s4.html
[Accessed February 23, 2016]

[2] Cisco IOS Image Verification [No Date]
http://www.cisco.com/c/en/us/about/security-center/ios-image-verification.html
[Accessed February 23, 2016]

[3] Image Verification [January 18, 2012]
http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_usr_cfg/configuration/12-4/sec-usr-cfg-12-4-book/sec-image-verifctn.html
[Accessed February 23, 2016]

[4] Cisco IOS Software Integrity Assurance [June 08, 2015]
http://www.cisco.com/web/about/security/intelligence/integrity-assurance.html
[Accessed February 23, 2016]

[5] Cisco IOS Shellcodes [2007]
https://www.blackhat.com/presentations/bh-usa-08/Chawdhary_Uppal/BH_US_08_Chawdhary_Uppal_Cisco_IOS_Shellcodes.pdf
[Accessed February 23, 2016]

[6] Securing Tool Command Language on Cisco IOS [No Date]
http://www.cisco.com/c/en/us/about/security-center/secure-tool-command-language.html
[Accessed February 23, 2016]

[7] Cisco IOS Resilient Configuration [October 19, 2009]
http://www.cisco.com/c/en/us/td/docs/ios/sec_user_services/configuration/guide/15_0s/sec_securing_user_services_15_0S_book/sec_resil_config.html
[Accessed February 23, 2016]

[8] Cisco Security Response: Rootkits on Cisco IOS Devices [April 9, 2014]
https://tools.cisco.com/security/center/content/CiscoSecurityResponse/cisco-sr-20080516-rootkits
[Accessed February 23, 2016]

[9] Support - Secure Publishing - Bulk Hash File Download Center [No Date]
http://www.cisco.com/c/en/us/support/web/tools/bulk-hash/index.html
[Accessed February 23, 2016]

[10] Loading and Maintaining System Images Configuration Guide [April 21, 2010]
http://www.cisco.com/c/en/us/td/docs/ios/fundamentals/configuration/guide/TIPs_Conversion/lmsi_15_1s_book/cf_dgtly_sgnd_sw.html
[Accessed February 23, 2016]

[11] Creating Core Dumps - Cisco [June 24, 2008]
http://www.cisco.com/c/en/us/support/docs/ios-nx-os-software/ios-software-releases-122-mainline/12687-appA.html
[Accessed February 23, 2016]

[12] Offline Analysis of IOS Image Integrity [February 23, 2012]
http://blogs.cisco.com/security/offline-analysis-of-ios-image-integrity
[Accessed February 23, 2016]

[13] Developments in Cisco IOS Forensics [February 2008]
http://www.recurity-labs.com/content/pub/RecurityLabs_Developments_in_IOS_Forensics.pdf
[Accessed February 23, 2016]

[14] Developments in Cisco IOS Forensics [August 2008]
http://www.recurity-labs.com/content/pub/RecurityLabs_Developments_in_IOS_Forensics_USA08.pdf
[Accessed February 23, 2016]

[15] Developments in Cisco IOS Forensics [January 2008]
https://www.blackhat.com/presentations/bh-dc-08/FX/Whitepaper/bh-dc-08-fx-WP.pdf
[Accessed February 23, 2016]

[16] Network Appliance Forensic Toolkit [December 2012]
http://blog.didierstevens.com/programs/network-appliance-forensic-toolkit/
[Accessed February 23, 2016]

[17] Cisco Incident Response - CIR [2012]
http://cir.recurity.com/
[Accessed February 23, 2016]

[18] CISCO IOS SHELLCODE: ALL-IN-ONE [2015]
http://2015.zeronights.org/assets/files/05-Nosenko.pdf
[Accessed February 23, 2016]

[19] Killing the myth of Cisco IOS rootkits: DIK (Da Ios rootkit) [May 2008]
http://www.coresecurity.com/files/attachments/Killing_the_myth_of_Cisco_IOS_rootkits.pdf
[Accessed February 23, 2016]

[20] CISCO IOS/IOS XE Risk Mitigation [October 2014]
http://cert.europa.eu/static/WhitePapers/CERT-EU-SWP_14_08_CISCO-Risk-Mitigation_1_5.pdf
[Accessed February 23, 2016]

[21] IOSTrojan: Who really owns your router? [August 4, 2009]
http://www.sans.org/reading-room/whitepapers/malicious/iostrojan-owns-router-33324
[Accessed February 23, 2016]