CHAPTER 59

Monitoring System Processes and Logs

This chapter provides details on monitoring the health of the switch. It includes the following sections:

Displaying System Processes

Use the **show processes** command to obtain general information about all processes (see Example 59-1 to Example 59-6).

Example 59-1 Displays System Processes

```
switch# show processes
PID    State  PC        Start_cnt    TTY   Process
-----  -----  --------  -----------  ----  -------------
  868      S  2ae4f33e            1     -  snmpd
  869      S  2acee33e            1     -  rscn
  870      S  2ac36c24            1     -  qos
  871      S  2ac44c24            1     -  port-channel
  872      S  2ac7a33e            1     -  ntp
    -     ER         -            1     -  mdog
    -     NR         -            0     -  vbuilder
```

Where:

- PID = process ID.
- State = process state.
  - D = uninterruptible sleep (usually I/O).
  - R = runnable (on run queue).
  - S = sleeping.
  - T = traced or stopped.

  - Z = defunct (“zombie”) process.
- NR = not running.
- ER = should be running but currently not-running.
- PC = current program counter in hex format.
- Start_cnt = number of times a process has been started (or restarted).
- TTY = terminal that controls the process. A hyphen usually means a daemon not running on any particular TTY.
- Process = name of the process.

Example 59-2 Displays CPU Utilization Information

```
switch# show processes cpu
PID    Runtime(ms)  Invoked   uSecs  1Sec   Process
-----  -----------  --------  -----  -----  -----------
  842         3807    137001     27    0.0  sysmgr
 1112         1220     67974     17    0.0  syslogd
 1269          220     13568     16    0.0  fcfwd
 1276         2901     15419    188    0.0  zone
 1277          738     21010     35    0.0  xbar_client
 1278         1159      6789    170    0.0  wwn
 1279          515     67617      7    0.0  vsan
```

Where:

- Runtime (ms) = CPU time the process has used, expressed in milliseconds.
- Invoked = number of times the process has been invoked.
- uSecs = microseconds of CPU time on average for each process invocation.
- 1Sec = CPU utilization in percentage for the last one second.

Example 59-3 Displays Process Log Information

```
switch# show processes log
Process           PID     Normal-exit  Stack-trace  Core     Log-create-time
----------------  ------  -----------  -----------  -------  ---------------
fspf              1339              N            Y        N  Jan  5 04:25
lcm               1559              N            Y        N  Jan  2 04:49
rib               1741              N            Y        N  Jan  1 06:05
```

Where:

- Normal-exit = whether or not the process exited normally.
- Stack-trace = whether or not there is a stack trace in the log.
- Core = whether or not there exists a core file.
- Log-create-time = when the log file got generated.

Example 59-4 Displays Detail Log Information About a Process

```
switch# show processes log pid 1339
Service: fspf
Description: FSPF Routing Protocol Application

Started at Sat Jan  5 03:23:44 1980 (545631 us)
Stopped at Sat Jan  5 04:25:57 1980 (819598 us)
```

```
Uptime: 1 hours 2 minutes 2 seconds

Start type: SRV_OPTION_RESTART_STATELESS (23)
Death reason: SYSMGR_DEATH_REASON_FAILURE_SIGNAL (2)
Exit code: signal 9 (no core)
CWD: /var/sysmgr/work

Virtual Memory:

    CODE      08048000 - 0809A100
    DATA      0809B100 - 0809B65C
    BRK       0809D988 - 080CD000
    STACK     7FFFFD20
    TOTAL     23764 KB

Register Set:

    EBX 00000005         ECX 7FFFF8CC         EDX 00000000
    ESI 00000000         EDI 7FFFF6CC         EBP 7FFFF95C
    EAX FFFFFDFE         XDS 8010002B         XES 0000002B
    EAX 0000008E (orig)  EIP 2ACE133E         XCS 00000023
    EFL 00000207         ESP 7FFFF654         XSS 0000002B

Stack: 1740 bytes. ESP 7FFFF654, TOP 7FFFFD20

0x7FFFF654: 00000000 00000008 00000003 08051E95 ................
0x7FFFF664: 00000005 7FFFF8CC 00000000 00000000 ................
0x7FFFF674: 7FFFF6CC 00000001 7FFFF95C 080522CD ........\....".."
0x7FFFF684: 7FFFF9A4 00000008 7FFFFC34 2AC1F18C ........4......*
```

Example 59-5 Displays All Process Log Details

```
switch# show processes log details
================================================
Service: snmpd
Description: SNMP Agent

Started at Wed Jan  9 00:14:55 1980 (597263 us)
Stopped at Fri Jan 11 10:08:36 1980 (649860 us)
Uptime: 2 days 9 hours 53 minutes 53 seconds

Start type: SRV_OPTION_RESTART_STATEFUL (24)
Death reason: SYSMGR_DEATH_REASON_FAILURE_SIGNAL (2)
Exit code: signal 6 (core dumped)
CWD: /var/sysmgr/work

Virtual Memory:

    CODE      08048000 - 0804C4A0
    DATA      0804D4A0 - 0804D770
    BRK       0804DFC4 - 0818F000
    STACK     7FFFFCE0
    TOTAL     26656 KB
...
```

Example 59-6 Displays Memory Information About Processes

```
switch# show processes memory
PID    MemAlloc  StackBase/Ptr      Process
-----  --------  -----------------  ----------------
 1277    120632  7ffffcd0/7fffefe4  xbar_client
 1278     56800  7ffffce0/7ffffb5c  wwn
```

```
1279   1210220  7ffffce0/7ffffbac  vsan
1293    386144  7fffcf0/7fffebd4  span
1294   1396892  7ffffce0/7fffdff4  snmpd
1295    214528  7fffcf0/7ffff904  rscn
1296     42064  7ffffce0/7ffffb5c  qos
```

Where:

- MemAlloc = total memory allocated by the process.
- StackBase/Ptr = process stack base and current stack pointer in hex format.

Displaying System Status

Use the **show system** command to display system-related status information (see Example 59-7 to Example 59-10.

Example 59-7 Displays Default Switch Port States

```
switch# show system default switchport
System default port state is down
System default trunk mode is on
```

Example 59-8 Displays Error Information for a Specified ID

```
switch# show system error-id 0x401D0019
Error Facility: module
Error Description: Failed to stop Linecard Async Notification.
```

Example 59-9 Displays the System Reset Information

```
switch# Show system reset-reason module 5
----- reset reason for module 5 -----
1) At 224801 usecs after Fri Nov 21 16:36:40 2003
    Reason: Reset Requested by CLI command reload
    Service:
    Version: 1.3(1)
2) At 922828 usecs after Fri Nov 21 16:02:48 2003
    Reason: Reset Requested by CLI command reload
    Service:
    Version: 1.3(1)
3) At 318034 usecs after Fri Nov 21 14:03:36 2003
    Reason: Reset Requested by CLI command reload
    Service:
    Version: 1.3(1)
4) At 255842 usecs after Wed Nov 19 00:07:49 2003
    Reason: Reset Requested by CLI command reload
    Service:
    Version: 1.3(1)
```

The **show system reset-reason** command displays the following information:

- In a Cisco MDS 9513 Director, the last four reset-reason codes for the supervisor module in slot 7 and slot 8 are displayed. If either supervisor module is absent, the reset-reason codes for that supervisor module are not displayed.

- In a Cisco MDS 9506 or Cisco MDS 9509 switch, the last four reset-reason codes for the supervisor module in slot 5 and slot 6 are displayed. If either supervisor module is absent, the reset-reason codes for that supervisor module are not displayed.
- In a Cisco MDS 9200 Series switch, the last four reset-reason codes for the supervisor module in slot 1 are displayed.
- The **show system reset-reason module** *number* command displays the last four reset-reason codes for a specific module in a given slot. If a module is absent, then the reset-reason codes for that module are not displayed.

Use the **clear system reset-reason** command to clear the reset-reason information stored in NVRAM and volatile persistent storage.

- In a Cisco MDS 9500 Series switch, this command clears the reset-reason information stored in NVRAM and volatile persistent storage in the active and standby supervisor modules.
- In a Cisco MDS 9200 Series switch, this command clears the reset-reason information stored in NVRAM and volatile persistent storage in the active supervisor module.

Example 59-10 Displays System Uptime

```
switch# show system uptime
Start Time: Sun Oct 13 18:09:23 2030
Up Time:    0 days, 9 hours, 46 minutes, 26 seconds
```

Use the **show system resources** command to display system-related CPU and memory statistics (see Example 59-11).

Example 59-11 Displays System-Related CPU and Memory Information

```
switch# show system resources
Load average:   1 minute: 0.43   5 minutes: 0.17   15 minutes: 0.11
Processes   :   100 total, 2 running
CPU states  :   0.0% user,   0.0% kernel,   100.0% idle
Memory usage:   1027628K total,    313424K used,    714204K free
                   3620K buffers,   22278K cache
```

Where:

- Load average—Displays the number of running processes. The average reflects the system load over the past 1, 5, and 15 minutes.
- Processes—Displays the number of processes in the system, and how many are actually running when the command is issued.
- CPU states—Displays the CPU usage percentage in user mode, kernel mode, and idle time in the last one second.
- Memory usage—Displays the total memory, used memory, free memory, memory used for buffers, and memory used for cache in KB. Buffers and cache are also included in the *used* memory statistics.

Core and Log Files

This section the following topics:

Displaying Core Status

Use the **show system cores** command to display the currently configured scheme for copying cores. See Examples 59-12 to 59-14.

Example 59-12 Displays the Status of System Cores

```
switch# show system cores
Transfer of cores is enabled
```

Example 59-13 Displays All Cores Available for Upload from the Active Supervisor Module

```
switch# show cores
Module-num  Process-name    PID     Core-create-time
----------  ------------    ---     ----------------
5           fspf            1524    Nov 9 03:11
6           fcc             919     Nov 9 03:09
8           acltcam         285     Nov 9 03:09
8           fib             283     Nov 9 03:08
```

Where `Module-num` shows the slot number on which the core was generated. In this example, the `fspf core` was generated on the active supervisor module (slot 5), `fcc` was generated on the standby supervisor module (slot 6), and `acltcam` and `fib` were generated on the switching module (slot 8).

Example 59-14 Displays Logs on the Local System

```
switch# show processes log
Process            PID     Normal-exit  Stack  Core   Log-create-time
---------------    ------  -----------  -----  -----  ---------------
ExceptionLog       2862             N      Y      N  Wed Aug  6 15:08:34 2003
acl                2299             N      Y      N  Tue Oct 28 02:50:01 2003
bios_daemon        2227             N      Y      N  Mon Sep 29 15:30:51 2003
capability         2373             N      Y      N  Tue Aug 19 13:30:02 2003
core-client        2262             N      Y      N  Mon Sep 29 15:30:51 2003
fcanalyzer         5623             N      Y      N  Fri Sep 26 20:45:09 2003
fcd                12996            N      Y      N  Fri Oct 17 20:35:01 2003
fcdomain           2410             N      Y      N  Thu Jun 12 09:30:58 2003
ficon              2708             N      Y      N  Wed Nov 12 18:34:02 2003
ficonstat          9640             N      Y      N  Tue Sep 30 22:55:03 2003
flogi              1300             N      Y      N  Fri Jun 20 08:52:33 2003
idehsd             2176             N      Y      N  Tue Jun 24 05:10:56 2003
lmgrd              2220             N      N      N  Mon Sep 29 15:30:51 2003
platform           2840             N      Y      N  Sat Oct 11 18:29:42 2003
port-security      3098             N      Y      N  Sun Sep 14 22:10:28 2003
port               11818            N      Y      N  Mon Nov 17 23:13:37 2003
rlir               3195             N      Y      N  Fri Jun 27 18:01:05 2003
rscn               2319             N      Y      N  Mon Sep 29 21:19:14 2003
securityd          2239             N      N      N  Thu Oct 16 18:51:39 2003
snmpd              2364             N      Y      N  Mon Nov 17 23:19:39 2003
span               2220             N      Y      N  Mon Sep 29 21:19:13 2003
syslogd            2076             N      Y      N  Sat Oct 11 18:29:40 2003
tcap               2864             N      Y      N  Wed Aug  6 15:09:04 2003
```

```
tftpd               2021            N       Y       N  Mon Sep 29 15:30:51 2003
vpm                 2930            N       N       N  Mon Nov 17 19:14:33 2003
```

Saving Cores

You can save cores (from the active supervisor module, the standby supervisor module, or any switching module) to an external CompactFlash (slot 0) or to a TFTP server in one of two ways:

- On demand—Copies a single file based on the provided process ID.
- Periodically—Copies core files periodically as configured by the user.

A new scheme overwrites any previously issued scheme. For example, if you perform another core log copy task, the cores are periodically saved to the new location or file.

Tip Be sure to create any required directory before performing this task. If the directory specified by this task does not exist, the switch software logs a system message each time a copy cores is attempted.

To copy the core and log files on demand, follow this step:

	Command	Purpose
Step 1	`switch# show cores`	
Step 2	`switch# copy core:7407 slot0:coreSample`	Copies the core file with the process ID 7407 as coreSample in slot 0.
	`switch# copy core://5/1524 tftp:/1.1.1.1/abcd`	Copies cores (if any) of a process with PID 1524 generated on slot 5[1] or slot 7[2] to the TFTP server at IPv4 address 1.1.1.1. **Note** You can also use IPv6 addresses to identify the TFTP server.

1. Cisco MDS 9506 or Cisco MDS 9509 switch
2. Cisco MDS 9513 Director

- If the core file for the specified process ID is not available, you see the following response:

```
switch# copy core:133 slot0:foo
No core file found with pid 133
```

- If two core files exist with the same process ID, only one file is copied:

```
switch# copy core:7407 slot0:foo1
2 core files found with pid 7407
Only "/isan/tmp/logs/calc_server_log.7407.tar.gz" will be copied to the destination.
```

To copy the core and log files periodically, follow these steps:

	Command	Purpose
Step 1	`switch# show system cores`	
Step 2	`switch# config t`	Enters configuration mode.

	Command	Purpose
Step 3	`switch(config)# system cores slot0:coreSample`	Copies the core file (coreSample) to slot 0.
	`switch(config)# system cores tftp:/1.1.1.1/abcd`	Copies the core file (abcd) in the specified directory on the TFTP server at IPv4 address 1.1.1.1. **Note** You can also use IPv6 addresses to identify the TFTP server.
	`switch(config)# no system cores`	Disables the core files copying feature.

Saving the Last Core to CompactFlash

This last core dump is automatically saved to CompactFlash in the /mnt/pss/ partition before the switchover or reboot occurs. Three minutes after the supervisor module reboots, the saved last core is restored from the Flash partition (/mnt/pss) back to its original RAM location. This restoration is a background process and is not visible to the user.

Tip The timestamp on the restored last core file displays the time when the supervisor booted up—not when the last core was actually dumped. To obtain the exact time of the last core dump, check the corresponding log file with the same PID.

To view the last core information, issue the **show cores** command in EXEC mode.

To view the time of the actual last core dump, issue the **show process log** command in EXEC mode.

Clearing the Core Directory

Use the **clear cores** command to clean out the core directory. The software keeps the last few cores per service and per slot and clears all other cores present on the active supervisor module.

```
switch# clear cores
```

First and Last Core

The First and last core feature uses the limited system resource and retains the most important core files. Generally, the first core and the most recently generated core have the information for debugging and, the First and last core feature tries to retain the first and the last core information.

If the core files are generated from active supervisor module, the number of core files for the service is defined in the service.conf file. There is no upper limit on the total number of core files in the active supervisor module. The defined number of core files work for every VDC.

To display the core files saved in the system, use the following commands:

Command	Purpose
switch# show cores	Displays all the core files saved in the default-VDC.
switch# show cores vdc-all	Displays all the core files saved in the system. The number of core files is defined in service.conf file.

First and Last Core Verification

You can view specific information about the saved core files. Example 59-15 to Example 59-16 provide further details on saved core files.

Example 59-15 Regular Service on Default-VDC on Local Node

For example, pixm crashes five times. The output of show cores vdc-all displays five core files. Three minutes later, the second oldest core file gets deleted to comply with the number of cores defined in the service.conf file.

```
switch# show cores vdc-all

VDC No Module-num       Process-name      PID     Core-create-time
------ ----------       ------------      ---     ----------------
1      5                pixm              4103    Jan 29 01:30
1      5                pixm              5105    Jan 29 01:32
1      5                pixm              5106    Jan 29 01:32
1      5                pixm              5107    Jan 29 01:33
1      5                pixm              5108    Jan 29 01:40

switch# show cores vdc-all

VDC No Module-num       Process-name      PID     Core-create-time
------ ----------       ------------      ---     ----------------
1      5                pixm              4103    Jan 29 01:30
1      5                pixm              5106    Jan 29 01:32
1      5                pixm              5107    Jan 29 01:33
1      5                pixm              5108    Jan 29 01:40
```

Example 59-16 Regular Service on vdc 2 on Active Supervisor Module

For example, there are five radius core files from vdc2 on the active supervisor module. The second and third oldest files get deleted to comply with the number of core files defined in the service.conf file.

```
switch# show cores vdc vdc2

VDC No Module-num       Process-name      PID     Core-create-time
------ ----------       ------------      ---     ----------------
2      5                radius            6100    Jan 29 01:47
2      5                radius            6101    Jan 29 01:55
2      5                radius            6102    Jan 29 01:55
2      5                radius            6103    Jan 29 01:55
2      5                radius            6104    Jan 29 01:57

switch# show cores vdc vdc2

VDC No Module-num       Process-name      PID     Core-create-time
------ ----------       ------------      ---     ----------------
2      5                radius            6100    Jan 29 01:47
```

```
2       5               radius          6103    Jan 29 01:55
2       5               radius          6104    Jan 29 01:57
```

Kernel Core Dumps

Caution Changes to the kernel cores should be made by an administrator or individual who is completely familiar with switch operations.

When a specific module's operating system (OS) crashes, it is sometimes useful to obtain a full copy of the memory image (called a *kernel core dump*) to identify the cause of the crash. When the module experiences a kernel core dump it triggers the proxy server configured on the supervisor. The supervisor sends the module's OS kernel core dump to the Cisco MDS 9000 System Debug Server. Similarly, if the supervisor OS fails, the supervisor sends its OS kernel core dump to the Cisco MDS 9000 System Debug Server.

Note The Cisco MDS 9000 System Debug Server is a Cisco application that runs on Linux. It creates a repository for kernel core dumps. You can download the Cisco MDS 9000 System Debug Server from the Cisco.com website at http://www.cisco.com/kobayashi/sw-center/sw-stornet.shtml.

Kernel core dumps are only useful to your technical support representative. The kernel core dump file, which is a large binary file, must be transferred to an external server that resides on the same physical LAN as the switch. The core dump is subsequently interpreted by technical personnel who have access to source code and detailed memory maps.

Tip Core dumps take up disk space on the Cisco MDS 9000 System Debug Server application. If all levels of core dumps (**level all** option) are configured, you need to ensure that a minimum of 1 GB of disk space is available on the Linux server running the Cisco MDS 9000 System Debug Server application to accept the dump. If the process does not have sufficient space to complete the generation, the module resets itself. All changes made to kernel cores are saved to the running configuration.

This section includes the following topics:

Configuring External Servers

To configure the external server using IPv4, follow these steps:

	Command	Purpose
Step 1	`switch# config terminal` `switch(config)#`	Enters configuration mode.
Step 2	`switch(config)# kernel core target 10.50.5.5` `succeeded`	Configures the external server's IPv4 address. **Note** IPv6 addresses are not supported for kernel core targets.

Configuring Module Parameters

To configure the module parameters, follow these steps:

	Command	Purpose
Step 1	`switch# config terminal` `switch(config)#`	Enters configuration mode.
Step 2	`switch(config)# kernel core module 5` `succeeded`	Configures kernel core generation for module 5.
	`switch(config)# kernel core module 5 level header` `succeeded`	Configures kernel core generation for module 5, and limits the generation to header-level cores.
Step 3	`switch(config)# kernel core limit 2` `succeeded`	Configures kernel core generations for two modules. The default is 1 module.

Displaying Kernel Core Information

All changes made to the kernel cores may be viewed using the **show running-config** command. Alternatively, use the **show kernel cores** command to view specific configuration changes (see Example 59-17 to Example 59-19).

Example 59-17 Displays the Core Limit

```
switch# show kernel core limit
2
```

Example 59-18 Displays the External Server

```
switch# show kernel core target
10.50.5.5
```

Example 59-19 Displays the Core Settings for the Specified Module

```
switch# show kernel core module 5
module 5 core is enabled
         level is header
         dst_ip is 10.50.5.5
         src_port is 6671
```

```
dst_port is 6666
dump_dev_name is eth1
dst_mac_addr is 00:00:0C:07:AC:01
```

Online System Health Management

The Online Health Management System (system health) is a hardware fault detection and recovery feature. It ensures the general health of switching, services, and supervisor modules in any switch in the Cisco MDS 9000 Family.

This section includes the following topics:

About Online System Health Management

The Online Health Management System (OHMS) is a hardware fault detection and recovery feature. It runs on all Cisco MDS switching, services, and supervisor modules and ensures the general health of any switch in the Cisco MDS 9000 Family. The OHMS monitors system hardware in the following ways:

- The OHMS component running on the active supervisor maintains control over all other OHMS components running on the other modules in the switch.
- The system health application running in the standby supervisor module only monitors the standby supervisor module—if that module is available in the HA standby mode. See the "HA Switchover Characteristics" section on page 9-2.

The OHMS application launches a daemon process in all modules and runs multiple tests on each module to test individual module components. The tests run at preconfigured intervals, cover all major fault points, and isolate any failing component in the MDS switch. The OHMS running on the active supervisor maintains control over all other OHMS components running on all other modules in the switch.

On detecting a fault, the system health application attempts the following recovery actions:

- Performs additional testing to isolate the faulty component

- Attempts to reconfigure the component by retrieving its configuration information from persistent storage.
- If unable to recover, sends Call Home notifications, system messages and exception logs; and shuts down and discontinues testing the failed module or component (such as an interface)
- Sends Call Home and system messages and exception logs as soon as it detects a failure.
- Shuts down the failing module or component (such as an interface).
- Isolates failed ports from further testing.
- Reports the failure to the appropriate software component.
- Switches to the standby supervisor module, if an error is detected on the active supervisor module and a standby supervisor module exists in the Cisco MDS switch. After the switchover, the new active supervisor module restarts the active supervisor tests.
- Reloads the switch if a standby supervisor module does not exist in the switch.
- Provides CLI support to view, test, and obtain test run statistics or change the system health test configuration on the switch.
- Performs tests to focus on the problem area.

Each module is configured to run the test relevant to that module. You can change the default parameters of the test in each module as required.

System Health Initiation

By default, the system health feature is enabled in each switch in the Cisco MDS 9000 Family.

To disable or enable this feature in any switch in the Cisco MDS 9000 Family, follow these steps:

	Command	Purpose
Step 1	`switch# config terminal` `switch(config)#`	Enters configuration mode.
Step 2	`switch(config)# no system health` `System Health is disabled.`	Disables system health from running tests in this switch.
	`switch(config)# system health` `System Health is enabled.`	Enables (default) system health to run tests in this switch.
Step 3	`switch(config)# no system health interface fc8/1` `System health for interface fc8/13 is disabled.`	Disables system health from testing the specified interface.
	`switch(config)# system health interface fc8/1` `System health for interface fc8/13 is enabled.`	Enables (default) system health to test for the specified interface.

Loopback Test Configuration Frequency

Loopback tests are designed to identify hardware errors in the data path in the module(s) and the control path in the supervisors. One loopback frame is sent to each module at a preconfigured frequency—it passes through each configured interface and returns to the supervisor module.

The loopback tests can be run at frequencies ranging from 5 seconds (default) to 255 seconds. If you do not configure the loopback frequency value, the default frequency of 5 seconds is used for all modules in the switch. Loopback test frequencies can be altered for each module.

To configure the frequency of loopback tests for all modules on a switch, follow these steps:

	Command	Purpose
Step 1	`switch# config terminal` `switch(config)#`	Enters configuration mode.
Step 2	`switch(config)# system health loopback frequency 50` `The new frequency is set at 50 Seconds.`	Configures the loopback frequency to 50 seconds. The default loopback frequency is 5 seconds. The valid range is from 5 to 255 seconds.

Loopback Test Configuration Frame Length

Loopback tests are designed to identify hardware errors in the data path in the module(s) and the control path in the supervisors. One loopback frame is sent to each module at a preconfigured size—it passes through each configured interface and returns to the supervisor module.

The loopback tests can be run with frame sizes ranging from 0 bytes to 128 bytes. If you do not configure the loopback frame length value, the switch generates random frame lengths for all modules in the switch (auto mode). Loopback test frame lengths can be altered for each module.

To configure the frame length for loopback tests for all modules on a switch, follow these steps:

	Command	Purpose
Step 1	`switch# config terminal` `switch(config)#`	Enters configuration mode.
Step 2	`switch(config)# system health loopback frame-length 128`	Configures the loopback frame length to 128 bytes. The valid range is 0 to 128 bytes.
Step 3	`switch(config)# system health loopback frame-length auto`	Configures the loopback frame length to automatically generate random lengths (default).

To verify the loopback frequency configuration, use the **show system health loopback frame-length** command.

```
switch# show system health loopback frame-length
Loopback frame length is set to auto-size between 0-128 bytes
```

Hardware Failure Action

The failure-action command controls the Cisco SAN-OS software from taking any action if a hardware failure is determined while running the tests.

By default, this feature is enabled in all switches in the Cisco MDS 9000 Family—action is taken if a failure is determined and the failed component is isolated from further testing.

Failure action is controlled at individual test levels (per module), at the module level (for all tests), or for the entire switch.

To configure failure action in a switch, follow these steps:

	Command	Purpose
Step 1	`switch# config terminal` `switch(config)#`	Enters configuration mode.
Step 2	`switch(config)# system health failure-action` `System health global failure action is now enabled.`	Enables the switch to take failure action (default).
Step 3	`switch(config)# no system health failure-action` `System health global failure action now disabled.`	Reverts the switch configuration to prevent failure action being taken.
Step 4	`switch(config)# system health module 1 failure-action` `System health failure action for module 1 is now enabled.`	Enables switch to take failure action for failures in module 1.
Step 5	`switch(config)# no system health module 1 loopback failure-action` `System health failure action for module 1 loopback test is now disabled.`	Prevents the switch from taking action on failures determined by the loopback test in module 1.

Test Run Requirements

Enabling a test does not guarantee that a test will run.

Tests on a given interface or module only run if you enable system health for all of the following items:

- The entire switch.
- The required module.
- The required interface.

Tip The test will not run if system health is disabled in any combination. If system health is disabled to run tests, the test status shows up as disabled.

Tip If the specific module or interface is enabled to run tests, but is not running the tests due to system health being disabled, then tests show up as enabled (not running).

Tests for a Specified Module

The system health feature in the SAN-OS software performs tests in the following areas:

- Active supervisor's in-band connectivity to the fabric.
- Standby supervisor's arbiter availability.
- Bootflash connectivity and accessibility on all modules.
- EOBC connectivity and accessibility on all modules.
- Data path integrity for each interface on all modules.
- Management port's connectivity.
- Caching Services Module (CSM) batteries (for temperature, age, full-charge capacity, (dis)charge ability and backup capability) and cache disks (for connectivity, accessibility and raw disk I/O).

- User-driven test for external connectivity verification, port is shut down during the test (Fibre Channel ports only).
- User-driven test for internal connectivity verification (Fibre Channel and iSCSI ports).

To perform the required test on a specific module, follow these steps:

	Command	Purpose
Step 1	`switch# config terminal` `switch(config)#`	Enters configuration mode.
	Note The following steps can be performed in any order.	
Step 2	`switch(config)# system health module 8 battery-charger` `battery-charger test is not configured to run on module 8.`	Enables the battery-charger test on both batteries in the CSM residing in slot 8. If the switch does not have a CSM in slot 8, this message is issued.
Step 3	`switch(config)# system health module 8 cache-disk` `cache-disk test is not configured to run on module 8.`	Enables the cache-disk test on both disks in the CSM residing in slot 8. If the switch does not have a CSM in slot 8, this message is issued.
	Note The various options for each test are described in the next step. Each command can be configured in any order. The various options are presented in the same step for documentation purposes.	
Step 4	`switch(config)# system health module 8 bootflash` `System health for module 8 Bootflash is already enabled.`	Enables the bootflash test on module in slot 8.
	`switch(config)# system health module 8 bootflash frequency 200` `The new frequency is set at 200 Seconds.`	Sets the new frequency of the bootflash test on module 8 to 200 seconds.
Step 5	`switch(config)# system health module 8 eobc` `System health for module 8 EOBC is now enabled.`	Enables the EOBC test on module in slot 8.
Step 6	`switch(config)# system health module 8 loopback` `System health for module 8 EOBC is now enabled.`	Enables the loopback test on module in slot 8.
Step 7	`switch(config)# system health module 5 management` `System health for module 8 EOBC is now enabled.`	Enables the management test on module in slot 5.

Clearing Previous Error Reports

You can clear the error history for Fibre Channel interfaces, iSCSI interfaces, an entire module, or one particular test for an entire module. By clearing the history, you are directing the software to retest all failed components that were previously excluded from tests.

If you previously enabled the failure-action option for a period of time (for example, one week) to prevent OHMS from taking any action when a failure is encountered and after that week you are now ready to start receiving these errors again, then you must clear the system health error status for each test.

Tip The management port test cannot be run on a standby supervisor module.

Use the EXEC-level **system health clear-errors** command at the interface or module level to erase any previous error conditions logged by the system health application. The **battery-charger**, the **bootflash**, the **cache-disk**, the **eobc**, the **inband**, the **loopback**, and the **mgmt** test options can be individually specified for a given module.

The following example clears the error history for the specified Fibre Channel interface:

```
switch# system health clear-errors interface fc 3/1
```

The following example clears the error history for the specified module:

```
switch# system health clear-errors module 3
```

The following example clears the management test error history for the specified module:

```
switch# system health clear-errors module 1 mgmt
```

Performing Internal Loopback Tests

You can run manual loopback tests to identify hardware errors in the data path in the switching or services modules, and the control path in the supervisor modules. Internal loopback tests send and receive FC2 frames to/from the same ports and provide the round trip time taken in microseconds. These tests are available for Fibre Channel, IPS, and iSCSI interfaces.

Use the EXEC-level **system health internal-loopback** command to explicitly run this test on demand (when requested by the user) within ports for the entire module.

```
switch# system health internal-loopback interface iscsi 8/1
Internal loopback test on interface iscsi8/1 was  successful.
Sent 1 received 1 frames
Round trip time taken is 79 useconds
```

Use the EXEC-level **system health internal-loopback** command to explicitly run this test on demand (when requested by the user) within ports for the entire module and override the frame count configured on the switch.

```
switch# system health internal-loopback interface iscsi 8/1 frame-count 20
Internal loopback test on interface iscsi8/1 was  successful.
Sent 1 received 1 frames
Round trip time taken is 79 useconds
```

Use the EXEC-level **system health internal-loopback** command to explicitly run this test on demand (when requested by the user) within ports for the entire module and override the frame length configured on the switch.

```
switch# system health internal-loopback interface iscsi 8/1 frame-count 32
Internal loopback test on interface iscsi8/1 was  successful.
Sent 1 received 1 frames
Round trip time taken is 79 useconds
```

Note If the test fails to complete successfully, the software analyzes the failure and prints the following error:

```
External loopback test on interface fc 7/2 failed. Failure reason: Failed to loopback,
analysis complete Failed device ID 3 on module 1
```

Performing External Loopback Tests

You can run manual loopback tests to identify hardware errors in the data path in the switching or services modules, and the control path in the supervisor modules. External loopback tests send and receive FC2 frames to/from the same port or between two ports.

You need to connect a cable (or a plug) to loop the Rx port to the Tx port before running the test. If you are testing to/from the same port, you need a special loop cable. If you are testing to/from different ports, you can use a regular cable. This test is only available for Fibre Channel interfaces.

Use the EXEC-level **system health external-loopback interface** *interface* command to run this test on demand for external devices connected to a switch that is part of a long-haul network.

```
switch# system health external-loopback interface fc 3/1
This will shut the requested interfaces Do you want to continue (y/n)?  [n] y
External loopback test on interface fc3/1 was  successful.
Sent 1 received 1 frames
```

Use the EXEC-level **system health external-loopback source** *interface* **destination interface** *interface* command to run this test on demand between two ports on the switch.

```
switch# system health external-loopback source interface fc 3/1 destination interface fc
3/2
This will shut the requested interfaces Do you want to continue (y/n)?  [n] y
External loopback test on interface fc3/1 and interface fc3/2 was  successful.
Sent 1 received 1 frames
```

Use the EXEC-level **system health external-loopback** *interface* **frame-count** command to run this test on demand for external devices connected to a switch that is part of a long-haul network and override the frame count configured on the switch.

```
switch# system health external-loopback interface fc 3/1 frame-count 10
This will shut the requested interfaces Do you want to continue (y/n)?  [n] y
External loopback test on interface fc3/1 was  successful.
Sent 1 received 1 frames
```

Use the EXEC-level **system health external-loopback** *interface* **frame-length** command to run this test on demand for external devices connected to a switch that is part of a long-haul network and override the frame length configured on the switch.

```
switch# system health external-loopback interface fc 3/1 frame-length 64
This will shut the requested interfaces Do you want to continue (y/n)?  [n] y
External loopback test on interface fc3/1 was  successful.
Sent 1 received 1 frames
```

Use the **system health external-loopback** *interface* **force** command to shut down the required interface directly without a back out confirmation.

```
switch# system health external-loopback interface fc 3/1 force
External loopback test on interface fc3/1 was  successful.
Sent 1 received 1 frames
```

Note If the test fails to complete successfully, the software analyzes the failure and prints the following error:

```
External loopback test on interface fc 7/2 failed. Failure reason: Failed to loopback,
analysis complete Failed device ID 3 on module 1
```

Performing Serdes Loopbacks

Serializer/Deserializer (serdes) loopback tests the hardware for a port. These tests are available for Fibre Channel interfaces.

Use the EXEC-level **system health serdes-loopback** command to explicitly run this test on demand (when requested by the user) within ports for the entire module.

```
switch# system health serdes-loopback interface fc 3/1
This will shut the requested interfaces Do you want to continue (y/n)?  [n] y
Serdes loopback test passed for module 3 port 1
```

Use the EXEC-level **system health serdes-loopback** command to explicitly run this test on demand (when requested by the user) within ports for the entire module and override the frame count configured on the switch.

```
switch# system health serdes-loopback interface fc 3/1 frame-count 10
This will shut the requested interfaces Do you want to continue (y/n)?  [n] y
Serdes loopback test passed for module 3 port 1
```

Use the EXEC-level **system health serdes-loopback** command to explicitly run this test on demand (when requested by the user) within ports for the entire module and override the frame length configured on the switch.

```
switch# system health serdes-loopback interface fc 3/1 frame-length 32
This will shut the requested interfaces Do you want to continue (y/n)?  [n] y
Serdes loopback test passed for module 3 port 1
```

Note If the test fails to complete successfully, the software analyzes the failure and prints the following error:

```
External loopback test on interface fc 3/1 failed. Failure reason: Failed to loopback,
analysis complete Failed device ID 3 on module 3
```

Interpreting the Current Status

The status of each module or test depends on the current configured state of the OHMS test in that particular module (see Table 59-1).

Table 59-1 OHMS Configured Status for Tests and Modules

Status	Description
Enabled	You have currently enabled the test in this module and the test is not running.
Disabled	You have currently disabled the test in this module.
Running	You have enabled the test and the test is currently running in this module.
Failing	This state is displayed if a failure is imminent for the test running in this module—possibility of test recovery exists in this state.
Failed	The test has failed in this module—and the state cannot be recovered.
Stopped	The test has been internally stopped in this module by the Cisco SAN-OS software.
Internal failure	The test encountered an internal failure in this module. For example, the system health application is not able to open a socket as part of the test procedure.
Diags failed	The startup diagnostics has failed for this module or interface.

Table 59-1 OHMS Configured Status for Tests and Modules (continued)

Status	Description
On demand	The system health external-loopback or the system health internal-loopback tests are currently running in this module. Only these two commands can be issued on demand.
Suspended	Only encountered in the MDS 9100 Series due to one oversubscribed port moving to a E or TE port mode. If one oversubscribed port moves to this mode, the other three oversubscribed ports in the group are suspended.

The status of each test in each module is visible when you display any of the **show system health** commands. See the "Displaying System Health" section on page 59-20.

Displaying System Health

Use the **show system health** command to display system-related status information (see Example 59-20 to Example 59-25).

Example 59-20 Displays the Current Health of All Modules in the Switch

```
switch# show system health

Current health information for module 2.

Test                       Frequency    Status          Action
------------------------------------------------------------------
Bootflash                  5 Sec        Running         Enabled
EOBC                       5 Sec        Running         Enabled
Loopback                   5 Sec        Running         Enabled
------------------------------------------------------------------

Current health information for module 6.

Test                       Frequency    Status          Action
------------------------------------------------------------------
InBand                     5 Sec        Running         Enabled
Bootflash                  5 Sec        Running         Enabled
EOBC                       5 Sec        Running         Enabled
Management Port            5 Sec        Running         Enabled
------------------------------------------------------------------
```

Example 59-21 Displays the Current Health of a Specified Module

```
switch# show system health module 8

Current health information for module 8.

Test                       Frequency    Status          Action
------------------------------------------------------------------
Bootflash                  5 Sec        Running         Enabled
EOBC                       5 Sec        Running         Enabled
Loopback                   5 Sec        Running         Enabled
------------------------------------------------------------------
```

Example 59-22 Displays Health Statistics for All Modules

```
switch# show system health statistics

Test statistics for module # 1
--------------------------------------------------------------------------------
Test Name           State            Freq(s)    Run    Pass    Fail CFail Errs
--------------------------------------------------------------------------------
Bootflash           Running             5s    12900   12900       0     0    0
EOBC                Running             5s    12900   12900       0     0    0
Loopback            Running             5s    12900   12900       0     0    0
--------------------------------------------------------------------------------

Test statistics for module # 3
--------------------------------------------------------------------------------
Test Name           State            Freq(s)    Run    Pass    Fail CFail Errs
--------------------------------------------------------------------------------
Bootflash           Running             5s    12890   12890       0     0    0
EOBC                Running             5s    12890   12890       0     0    0
Loopback            Running             5s    12892   12892       0     0    0
--------------------------------------------------------------------------------

Test statistics for module # 5
--------------------------------------------------------------------------------
Test Name           State            Freq(s)    Run    Pass    Fail CFail Errs
--------------------------------------------------------------------------------
InBand              Running             5s    12911   12911       0     0    0
Bootflash           Running             5s    12911   12911       0     0    0
EOBC                Running             5s    12911   12911       0     0    0
Management Port     Running             5s    12911   12911       0     0    0
--------------------------------------------------------------------------------

Test statistics for module # 6
--------------------------------------------------------------------------------
Test Name           State            Freq(s)    Run    Pass    Fail CFail Errs
--------------------------------------------------------------------------------
InBand              Running             5s    12907   12907       0     0    0
Bootflash           Running             5s    12907   12907       0     0    0
EOBC                Running             5s    12907   12907       0     0    0
--------------------------------------------------------------------------------

Test statistics for module # 8
--------------------------------------------------------------------------------
Test Name           State            Freq(s)    Run    Pass    Fail CFail Errs
--------------------------------------------------------------------------------
Bootflash           Running             5s    12895   12895       0     0    0
EOBC                Running             5s    12895   12895       0     0    0
Loopback            Running             5s    12896   12896       0     0    0
--------------------------------------------------------------------------------
```

Example 59-23 Displays Statistics for a Specified Module

```
switch# show system health statistics module 3

Test statistics for module # 3
--------------------------------------------------------------------------------
Test Name           State            Freq(s)    Run    Pass    Fail CFail Errs
--------------------------------------------------------------------------------
Bootflash           Running             5s    12932   12932       0     0    0
EOBC                Running             5s    12932   12932       0     0    0
Loopback            Running             5s    12934   12934       0     0    0
--------------------------------------------------------------------------------
```

Example 59-24 Displays Loopback Test Statistics for the Entire Switch

```
switch# show system health statistics loopback
-------------------------------------------------------------------
Mod Port Status                Run      Pass      Fail   CFail Errs
  1   16 Running             12953     12953         0       0    0
  3   32 Running             12945     12945         0       0    0
  8    8 Running             12949     12949         0       0    0
-------------------------------------------------------------------
```

Example 59-25 Displays Loopback Test Statistics for a Specified Interface

```
switch# show system health statistics loopback interface fc 3/1
-------------------------------------------------------------------
Mod Port Status                Run      Pass      Fail   CFail Errs
  3    1 Running                 0         0         0       0    0
-------------------------------------------------------------------
```

Note Interface-specific counters will remain at zero unless the module-specific loopback test reports errors or failures.

Example 59-26 Displays the Loopback Test Time Log for All Modules

```
switch# show system health statistics loopback timelog
-------------------------------------------------------------------
Mod         Samples     Min(usecs)     Max(usecs)     Ave(usecs)
  1            1872            149            364            222
  3            1862            415            743            549
  8            1865            134            455            349
-------------------------------------------------------------------
```

Example 59-27 Displays the Loopback Test Time Log for a Specified Module

```
switch# show system health statistics loopback module 8 timelog
-------------------------------------------------------------------
Mod         Samples     Min(usecs)     Max(usecs)     Ave(usecs)
  8            1867            134            455            349
-------------------------------------------------------------------
```

On-Board Failure Logging

The Generation 2 Fibre Channel switching modules provide the facility to log failure data to persistent storage, which can be retrieved and displayed for analysis. This on-board failure logging (OBFL) feature stores failure and environmental information in nonvolatile memory on the module. The information will help in post-mortem analysis of failed cards.

This section includes the following topics:

About OBFL

OBFL data is stored in the existing CompactFlash on the module. OBFL uses the persistent logging (PLOG) facility available in the module firmware to store data in the CompactFlash. It also provides the mechanism to retrieve the stored data.

The data stored by the OBFL facility includes the following:

- Time of initial power-on
- Slot number of the card in the chassis
- Initial temperature of the card
- Firmware, BIOS, FPGA, and ASIC versions
- Serial number of the card
- Stack trace for crashes
- CPU hog information
- Memory leak information
- Software error messages
- Hardware exception logs
- Environmental history
- OBFL specific history information
- ASIC interrupt and error statistics history
- ASIC register dumps

Configuring OBFL for the Switch

To configure OBFL for all the modules on the switch, follow these steps

	Command	Purpose
Step 1	`switch# config terminal` `switch(config)#`	Enters configuration mode.

	Command	Purpose
Step 2	`switch(config)# hw-module logging onboard`	Enables all OBFL features.
	`switch(config)# hw-module logging onboard cpu-hog`	Enables the OBFL CPU hog events.
	`switch(config)# hw-module logging onboard environmental-history`	Enables the OBFL environmental history.
	`switch(config)# hw-module logging onboard error-stats`	Enables the OBFL error statistics.
	`switch(config)# hw-module logging onboard interrupt-stats`	Enables the OBFL interrupt statistics.
	`switch(config)# hw-module logging onboard mem-leak`	Enables the OBFL memory leak events.
	`switch(config)# hw-module logging onboard miscellaneous-error`	Enables the OBFL miscellaneous information.
	`switch(config)# hw-module logging onboard obfl-log`	Enables the boot uptime, device version, and OBFL history.
	`switch(config)# no hw-module logging onboard`	Disables all OBFL features.

Use the **show logging onboard status** command to display the configuration status of OBFL.

```
switch# show logging onboard status

    Switch OBFL Log:                                        Enabled

    Module:  6 OBFL Log:                                    Enabled
    error-stats                                             Enabled
    exception-log                                           Enabled
    miscellaneous-error                                     Enabled
    obfl-log (boot-uptime/device-version/obfl-history)      Enabled
    system-health                                           Enabled
    stack-trace                                             Enabled
```

Configuring OBFL for a Module

To configure OBFL for specific modules on the switch, follow these steps

	Command	Purpose
Step 1	`switch# config terminal` `switch(config)#`	Enters configuration mode.

	Command	Purpose
Step 2	`switch(config)# hw-module logging onboard module 1`	Enables all OBFL features on a module.
	`switch(config)# hw-module logging onboard module 1 cpu-hog`	Enables the OBFL CPU hog events on a module.
	`switch(config)# hw-module logging onboard module 1 environmental-history`	Enables the OBFL environmental history on a module.
	`switch(config)# hw-module logging onboard module 1 error-stats`	Enables the OBFL error statistics on a module.
	`switch(config)# hw-module logging onboard module 1 interrupt-stats`	Enables the OBFL interrupt statistics on a module.
	`switch(config)# hw-module logging onboard module 1 mem-leak`	Enables the OBFL memory leak events on a module.
	`switch(config)# hw-module logging onboard module 1 miscellaneous-error`	Enables the OBFL miscellaneous information on a module.
	`switch(config)# hw-module logging onboard module 1 obfl-log`	Enables the boot uptime, device version, and OBFL history on a module.
	`switch(config)# no hw-module logging onboard module 1`	Disables all OBFL features on a module.

Use the **show logging onboard status** command to display the configuration status of OBFL.

```
switch# show logging onboard status

    Switch OBFL Log:                                        Enabled

    Module:  6 OBFL Log:                                    Enabled
    error-stats                                             Enabled
    exception-log                                           Enabled
    miscellaneous-error                                     Enabled
    obfl-log (boot-uptime/device-version/obfl-history)      Enabled
    system-health                                           Enabled
    stack-trace                                             Enabled
```

Displaying OBFL Logs

To display OBFL information stored in CompactFlash on a module, use the following commands:

Command	Purpose
show logging onboard boot-uptime	Displays the boot and uptime information.
show logging onboard cpu-hog	Displays information for CPU hog events.
show logging onboard device-version	Displays device version information.
show logging onboard endtime	Displays OBFL logs to an end time.
show logging onboard environmental-history	Displays environmental history.
show logging onboard error-stats	Displays error statistics.
show logging onboard exception-log	Displays exception log information.
show logging onboard interrupt-stats	Displays interrupt statistics.
show logging onboard mem-leak	Displays memory leak information.
show logging onboard miscellaneous-error	Displays miscellaneous error information.
show logging onboard module *slot*	Displays OBFL information for a specific module.
show logging onboard obfl-history	Displays history information.
show logging onboard register-log	Displays register log information.
show logging onboard stack-trace	Displays kernel stack trace information.
show logging onboard starttime	Displays OBFL logs from a specified start time.
show logging onboard system-health	Displays system health information.

Default Settings

Table 59-2 lists the default system health and log settings.

Table 59-2 Default System Health and Log Settings

Parameters	Default
Kernel core generation	One module.
System health	Enabled.
Loopback frequency	5 seconds.
Failure action	Enabled.