System Status Monitoring

This chapter provides details on monitoring the health of the switch.

Feature History for System Status Monitoring

Table 1: Feature History for System Status Monitoring , on page 1 lists the release history for this feature. Only features that were introduced or modified in Release 3.x or a later release appear in the table.

Table 1: Feature History for System Status Monitoring

Feature Name	Releases	Feature Information
Kernel Core Logging	8.4(2c)	Core files are created when NX-OS encounters an unrecoverable fault. The core files can be used by Cisco to diagnose the fault.
Common Information Model	3.3(1a)	Added commands for displaying Common Information Model.
On-line system health maintenance (OHMS) enhancements	3.0(1)	Includes the following OHMS enhancements: • Configuring the global frame length for loopback test for all modules on the switch. • Specifying frame count and frame length on for the loopback test on a specific module. • Configuring source and destination ports for external loopback tests. • Providing serdes loopback test to check hardware.

Feature Name	Releases	Feature Information
On-board failure logging (OBFL)	3.0(1)	Describes OBFL, how to configure it for Generation 2 modules, and how to display the log information.

Information About System Status Monitoring

Online Health Management System

The Online Health Management System (OHMS) (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 series.

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.

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.

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.

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.

Hardware Failure Action

The failure-action command controls the Cisco NX-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.

Performing Test Run Requirements

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

Tests on a specific 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 NX-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.
- 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).

Note In Cisco MDS 9700 Series Switches, iSCSI ports are not applicable.

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.

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 2: OHMS Configured Status for Tests and Modules , on page 4).

Table 2: 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 NX-OS software.

Status	Description
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.
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 , on page 21.

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.

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

Core Files

Core files are created when NX-OS encounters an unrecoverable fault. These are a bundle of files in *tar.gz* format and can be used by Cisco to diagnose the fault.

NX-OS can generate process and kernel core file from both supervisor and modules. Process core files are uploaded from the module they occur on to the active supervisor at the time of the fault. The core files are volatile and will be lost if the supervisor resets. Kernel core files are stored on the supervisor they were created on and are retained after a supervisor reset.

First and Last Core

Generally, the first and the most recent cores that are generated by a process have the most useful information for debugging. If the core files are generated on the active supervisor module, the first and last core feature automatically deletes the intermediate cores if a new one is generated for the same process to conserve space in the core repository.

Default Settings

Table 3: Default System Status Monitoring, on page 6 lists the default settings.

Table 3: Default System Status Monitoring

Parameters	Default
Kernel core collection	Disabled
System health	Enabled
Loopback frequency	5 seconds
Failure action	Enabled

Configuring System Health

The Online Health Management System (OHMS) (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.

Task Flow for Configuring System Health

Follow these steps to configure system health:

Procedure

Step 1 Enable System Health Initiation.

Step 2 Configure Loopback Test Configuration Frequency.

Step 3 Configure Loopback Test Configuration Frame Length.

Step 4 Configure Hardware Failure Action.

Step 5 Perform Test Run Requirements.

Step 6 Clear Previous Error Reports.

Step 7 Perform Internal Loopback Tests.

Step 8 Perform External Loopback Tests.

Step 9 Perform Serdes Loopbacks.

Configuring 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:

Procedure

Step 1 switch# **configure terminal**

Enters configuration mode.

Step 2 switch(config)# **no system health**

System Health is disabled.

Disables system health from running tests in this switch.

Step 3 switch(config)# **system health**

System Health is enabled.

Enables (default) system health to run tests in this switch.

Step 4 switch(config)# **no system health interface fc8/1**

System health for interface fc8/13 is disabled.

Disables system health from testing the specified interface.

Step 5 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.

Configuring Loopback Test Configuration Frequency

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

Procedure

Step 1 switch# **configure terminal**

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.

Configuring Loopback Test Configuration Frame Length

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

Procedure

Step 1 switch# **configure terminal**

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).

Configuring Hardware Failure Action

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

Procedure

Step 1 switch# **configure terminal**

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.

Performing Test Run Requirements

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

Procedure

Step 1 switch# **configure terminal**

Enters configuration mode.

Note The following steps can be performed in any order.

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 2 switch(config)# **system health module 8 bootflash**

Enables the bootflash test on module in slot 8.

Step 3 switch(config)# **system health module 8 bootflash frequency 200**

Sets the new frequency of the bootflash test on module 8 to 200 seconds.

Step 4 switch(config)# **system health module 8 eobc**

Enables the EOBC test on module in slot 8.

Step 5 switch(config)# **system health module 8 loopback**

Enables the loopback test on module in slot 8.

Step 6 switch(config)# **system health module 5 management**

Enables the management test on module in slot 5.

Clearing Previous Error Reports

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 **bootflash**, 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 and 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 and 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 and from the same port, you need a special loop cable. If you are testing to and 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.

Configuring On-Board Failure Logging

Each hardware module logs failure data to on-module 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.

Note It is recommended to clear statistics for all modules and switches before initiating In-Service Software Upgrade (ISSU). If hardware statistics are not cleared before initiating ISSU, you may see duplicate OBFL entries in system logs.

Configuring OBFL for a Switch

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

Procedure

Step 1 switch# **configure terminal**

Enters configuration mode.

Step 2 switch(config)# **hw-module logging onboard**

Enables all OBFL features.

Note This CLI only enable OBFL features that are disabled by no hw-module logging onboard command. For OBFL features that were individually disabled, please enable those using hw-module logging onboard obfl-feature command.

Step 3 switch(config)# **hw-module logging onboard cpu-hog**

Enables the OBFL CPU hog events.

Step 4 switch(config)# **hw-module logging onboard environmental-history**

Enables the OBFL environmental history.

Step 5 switch(config)# **hw-module logging onboard error-stats**

Enables the OBFL error statistics.

Step 6 switch(config)# **hw-module logging onboard interrupt-stats**

Enables the OBFL interrupt statistics.

Step 7 switch(config)# **hw-module logging onboard mem-leak**

Enables the OBFL memory leak events.

Step 8 switch(config)# **hw-module logging onboard miscellaneous-error**

Enables the OBFL miscellaneous information.

Step 9 switch(config)# **hw-module logging onboard obfl-log**

Enables the boot uptime, device version, and OBFL history.

Step 10 switch(config)# **no hw-module logging onboard**

Disables all OBFL features.

Configuring OBFL for a Module

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

Procedure

Step 1 switch# **configure terminal**

Enters configuration mode.

Step 2 switch(config)# **hw-module logging onboard module 1**

Enables all OBFL features on a module.

Step 3 switch(config)# **hw-module logging onboard module 1 cpu-hog**

Enables the OBFL CPU hog events on a module.

Step 4 switch(config)# **hw-module logging onboard module 1 environmental-history**

Enables the OBFL environmental history on a module.

Step 5 switch(config)# **hw-module logging onboard module 1 error-stats**

Enables the OBFL error statistics on a module.

Step 6 switch(config)# **hw-module logging onboard module 1 interrupt-stats**

Enables the OBFL interrupt statistics on a module.

Step 7 switch(config)# **hw-module logging onboard module 1 mem-leak**

Enables the OBFL memory leak events on a module.

Step 8 switch(config)# **hw-module logging onboard module 1 miscellaneous-error**

Enables the OBFL miscellaneous information on a module.

Step 9 switch(config)# **hw-module logging onboard module 1 obfl-log**

Enables the boot uptime, device version, and OBFL history on a module.

Step 10 switch(config)# **no hw-module logging onboard module 1**

Disables all OBFL features on a module.

Clearing the Module Counters

Note The module counters cannot be cleared using Device Manager or DCNM-SAN.

To reset the module counters, follow these steps:

Procedure

Step 1 switch# **attach module 1**

ModuleX#

Attaches module 1 to the chasiss.

Step 2 ModuleX# **clear asic-cnt all**

Clears the counters for all the devices in the module.

Step 3 ModuleX# **clear asic-cnt list-all-devices**

ModuleX# **clear asic-cnt device-id** *device-id*

Clears the counters for only the specified device ID. The device ID can vary from 1 through 255.

Resetting Counters for All Modules

To reset the counters for all the modules, follow these steps:

Procedure

switch# **debug system internal clear-counters all**

Clears the counters for all the modules in the switch.

Configuring Alerts, Notifications, and Monitoring of Counters

This section provides information on how to configure alerts, notification, and monitor counters.

Monitoring the CPU Utilization

To display the system CPU utilization, use the **show processes cpu** command.

This example shows how to display processes and CPU usage in the current VDC:

```
switch# show processes cpu
PID    Runtime(ms)  Invoked   uSecs  1Sec    Process
-----  -----------  --------  -----  ------  -----------
    4       386829  67421866      5    0.9%  ksoftirqd/0
3667       270567    396229    682    9.8%  syslogd
3942          262       161   1632    7.8%  netstack
4006    106999945  354495641    301   28.2%  snmpd
4026      4454796    461564   9651    0.9%  sac_usd
4424        84187    726180    115    0.9%  vpc
4426       146378    919073    159    0.9%  tunnel
CPU util  :   25.0% user,   30.5% kernel,   44.5% idle
```

Obtaining RAM Usage Information

You can obtain the processor RAM usage by using this SNMP variable: ceExtProcessorRam.

```
ceExtProcessorRam OBJECT-TYPE
    SYNTAX  Unsigned32
    UNITS   "bytes"
        MAX-ACCESS  read-only
        STATUS  current
        DESCRIPTION
        "Total number of bytes of RAM available on the
         Processor."
        ::= { ceExtPhysicalProcessorEntry 1 }
```

Monitoring Rx and Tx Traffic Counters

When monitoring Rx and Tx traffic counters, you should include the Rx counter OID:

```
ifHCInOctets
```

Monitoring Status of Interfaces

To monitor status of interfaces, use the IETF extended-linkDown trap, which has ifAlias (this trap can set interface description) and ifDescr, which shows port name in the ASCII format as shown below:

```
switch (config)# snmp-server enable traps link
  cieLinkDown                 Cisco extended link state down notification
  cieLinkUp                   Cisco extended link state up notification
  cisco-xcvr-mon-status-chg   Cisco interface transceiver monitor status change
                             notification
  delayed-link-state-change  Delayed link state change
  extended-linkDown           IETF extended link state down notification
  extended-linkUp             IETF extended link state up notification
  linkDown                   IETF Link state down notification
  linkUp                      IETF Link state up notification
switch (config)#
```

The following is an example of the trap:

```
[+]          10           16:41:39.79         IF-MIB:linkDown trap:SNMPv2c  from
```

```
[172.25.234.200  Port: 162 Community: public]
SNMPv2-MIB:sysUpTime.0 :  (35519336)      Syntax: TimeTicks
SNMPv2-MIB:snmpTrapOID.0 :  (IF-MIB:linkDown)      Syntax: ObjectID
IF-MIB:ifIndex.440414208 :   (440414208)  Syntax: INTEGER,  Instance IDs: (440414208)
IF-MIB:ifAdminStatus.440414208 :   (down)  Syntax: INTEGER,  Instance IDs: (440414208)
IF-MIB:ifOperStatus.440414208 :   (down)  Syntax: INTEGER,  Instance IDs: (440414208)
IF-MIB:ifDescr.440414208 :   (Ethernet9/4)  Syntax: RFC1213-MIB:DisplayString,  Instance
IDs: (440414208)
IF-MIB:ifAlias.440414208 :   (eth9/4)  Syntax: SNMPv2-TC:DisplayString,  Instance IDs:
(440414208)
SNMPv2-MIB:snmpTrapEnterprise.0 :  (IF-MIB:linkDown)      Syntax: ObjectID
```

Monitoring Transceiver Thresholds

Use the cisco-xcvr-mon-status-chg trap way to monitor digital diagnostics statistics for thresholds as shown below:

```
switch (config)# snmp-server enable traps link cisco-xcvr-mon-status-chg
switch (config)#
```

The trap MIB is as show below:

```
cIfXcvrMonStatusChangeNotif NOTIFICATION-TYPE
    OBJECTS         {
                        ifName,
                        cIfXcvrMonDigitalDiagTempAlarm,
                        cIfXcvrMonDigitalDiagTempWarning,
                        cIfXcvrMonDigitalDiagVoltAlarm,
                        cIfXcvrMonDigitalDiagVoltWarning,
                        cIfXcvrMonDigitalDiagCurrAlarm,
                        cIfXcvrMonDigitalDiagCurrWarning,
                        cIfXcvrMonDigitalDiagRxPwrAlarm,
                        cIfXcvrMonDigitalDiagRxPwrWarning,
                        cIfXcvrMonDigitalDiagTxPwrAlarm,
                        cIfXcvrMonDigitalDiagTxPwrWarning,
                        cIfXcvrMonDigitalDiagTxFaultAlarm
                    }
    STATUS          current
```

This example shows how to display transceiver details:

```
switch(config)# show interface ethernet 1/17 transceiver details
Ethernet1/17
    transceiver is present
    type is 10Gbase-SR
    name is CISCO-AVAGO
    part number is SFBR-7702SDZ
    revision is G2.3
    serial number is AGA1427618P
    nominal bitrate is 10300 MBit/sec
    Link length supported for 50/125um OM2 fiber is 82 m
    Link length supported for 62.5/125um fiber is 26 m
    Link length supported for 50/125um OM3 fiber is 300 m
    cisco id is --
    cisco extended id number is 4
           SFP Detail Diagnostics Information (internal calibration)
  ----------------------------------------------------------------------------
                Current              Alarms                  Warnings
                Measurement     High        Low         High          Low
```

```
  --------------------------------------------------------------------------------
  Temperature   27.65 C        75.00 C     -5.00 C     70.00 C        0.00 C
  Voltage        3.29 V         3.63 V      2.97 V      3.46 V        3.13 V
  Current        5.42 mA       10.50 mA     2.50 mA    10.50 mA       2.50 mA
  Tx Power      -2.51 dBm       1.69 dBm  -11.30 dBm   -1.30 dBm     -7.30 dBm
  Rx Power      -2.64 dBm       1.99 dBm  -13.97 dBm   -1.00 dBm     -9.91 dBm
  Transmit Fault Count = 0
  --------------------------------------------------------------------------------
  Note: ++  high-alarm; +  high-warning; --  low-alarm; -  low-warning
switch(config)#
```

Configuring Supervisor Switchover Notification

The supervisor switchover notification can be monitored by listening for the ciscoRFSwactNotif trap:

```
  ciscoRFSwactNotif NOTIFICATION-TYPE
  OBJECTS {
  cRFStatusUnitId,
  sysUpTime,
  cRFStatusLastSwactReasonCode
}
```

Configuring a Counter to Include CRC and FCS Errors

You can include CRC and FCS errors of interfaces by polling dot3StatsFCSErrors counter as shown in the following example:

dot3StatsFCSErrors Counter32

```
      Dot3StatsEntry ::= SEQUENCE {
  dot3StatsIndex                    InterfaceIndex,
  dot3StatsAlignmentErrors          Counter32,
  dot3StatsFCSErrors                Counter32,
  dot3StatsSingleCollisionFrames    Counter32,
  dot3StatsMultipleCollisionFrames  Counter32,
  dot3StatsSQETestErrors            Counter32,
  dot3StatsDeferredTransmissions    Counter32,
  dot3StatsLateCollisions           Counter32,
  dot3StatsExcessiveCollisions      Counter32,
  dot3StatsInternalMacTransmitErrors Counter32,
  dot3StatsCarrierSenseErrors       Counter32,
  dot3StatsFrameTooLongs            Counter32,
  dot3StatsInternalMacReceiveErrors  Counter32,
  dot3StatsEtherChipSet             OBJECT IDENTIFIER,
  dot3StatsSymbolErrors             Counter32,
  dot3StatsDuplexStatus             INTEGER,
  dot3StatsRateControlAbility       TruthValue,
  dot3StatsRateControlStatus        INTEGER
}
```

Configuring Call Home for Alerts

The Call Home feature enables you to receive a Call Home email when exceptions occur in the system. Use the following CLI or SNMP to set up the Call Home configurations and to enable all alert-groups:

```
switch (config)# callhome
switch-FC-VDC(config-callhome)# destination-profile full-txt-destination alert-group
```

```
All                   This alert group consists of all of the callhome
                      messages
  Cisco-TAC           Events which are meant for Cisco TAC only
  Configuration       Events related to Configuration
  Diagnostic          Events related to Diagnostic
  EEM                 EEM events
  Environmental       Power,fan,temperature related events
  Inventory           Inventory status events
  License             Events related to licensing
  Linecard-Hardware   Linecard related events
  Supervisor-Hardware Supervisor related events
  Syslog-group-port   Events related to syslog messages filed by port manager
  System              Software related events
  Test                User generated test events
switch-FC-VDC(config-callhome)#
```

Monitoring User Authentication Failures

You can monitor any user authentication failures by listening the authenticationFailure trap:

```
SNMPv2-MIB: authenticationFailure trap
```

Configuring Cores

Core files can be saved either manually by a user or automatically at the time of the fault. If a core file is created, preserve it by copying it to nonvolatile file space (such as to a host) and report it to Cisco for diagnosis.

Cores may be copied multiple times. Both IPv4, IPv6, and many protocols are supported for copying cores to file space on remote hosts. This includes passwordless SSH which is convenient for automatic copying in secure environments. For more information about configuring passwordless access to remote hosts, see the 'Passwordless File Copy and SSH' section in the 'Configuring SSH Services and Telnet' chapter of Cisco MDS 9000 Series Security Configuration Guide, Release 8.x.

There is no upper limit on the total number of core files in the active supervisor module.

Tip Before copying a core, ensure that you create the destination directory with write permission for the user.

Configuring Kernel Core Collection

To configure kernel core collection, follow these steps:

Procedure

Step 1 switch# **configure terminal**

Enters configuration mode.

Step 2 switch(config)# **system kernel core**

Enables collection of kernel core if there is a kernel crash.

Step 3 switch(config)# **no system kernel core**

(Optional) Disables collection of kernel cores.

Copying Cores Manually

The supported on-switch destination is slot0. Supported protocols to transfer cores to remote destinations are TFTP, SFTP, and SCP.

To configure saving cores manually, follow these steps:

Procedure

switch# **copy** *core://module/process-id[/instance] destination://[[user@]host/][directory]*

Copies a core of process to the specified location.

Copying Cores Automatically

Supported on-switch destinations are bootflash, slot0, and usb1. Supported protocols to transfer cores to remote destinations are HTTP, HTTPS, TFTP, FTP, SFTP, and SCP.

To configure saving cores automatically, follow these steps:

Procedure

Step 1 switch# **configure**

Enters configuration mode.

Step 2 switch(config)# **system cores** *destination://[[user@]host/][directory]*

Save core files to the specified destination as soon as they are created.

Step 3 switch(config)# **no system cores**

(Optional) Disables saving core files automatically.

Deleting Cores

Core files are not automatically deleted after copying. After a core is copied, delete it from the switch core repository to reclaim the space and report it to Cisco support for analysis.

Use the **clear core_file** command to delete a single core from the switch core repository.

switch# **clear core_file module** *module* **pid** *pid*

Use the **clear cores** command to clear all cores in the switch core repository.

switch# **clear cores**

Example: Configuring Cores

The following example copies a core of process with PID 1524 generated on slot 5 to the *cores* directory on a host with HTTPS as user *mdsadmin*:

```
switch# copy core://5/1524 https://mdsadmin@192.168.1.2/cores
```

The following example automatically copies any core files immediately after they are created to the /tftpboot/cores directory on a host with SCP as user mdsadmin. Configure passwordless SSH first for this to work.

```
switch# configure
switch(config)# system cores scp://mdsadmin@192.168.1.2/tftpboot/cores
```

The following example deletes the core generated from module 1 for the process with PID 1234.

```
switch# clear core_file module 1 pid 1234
```

Verifying System Status Monitoring Configuration

To display the system status monitoring configuration information, perform one of the following tasks:

Displaying System Health

Use the **show system health** command to display system-related status information (see Current Health of All Modules in the Switch, on page 21 to Loopback Test Time Log for a Specified Module, on page 24).

Current Health of All Modules in the Switch

The following example 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
----------------------------------------------------------------
```

Current Health of a Specified Module

The following example 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
----------------------------------------------------------------
```

Health Statistics for All Modules

The following example displays health statistics for all modules:

```
switch# show system health statistics
Test statistics for module # 1
-------------------------------------------------------------------------------
Test Name           State            Frequency 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            Frequency 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            Frequency 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            Frequency 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            Frequency 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
----------------------------------------------------------------------------
```

Displays Statistics for a Specified Module

The following example displays statistics for a specified module:

```
switch# show system health statistics module 3
Test statistics for module # 3
----------------------------------------------------------------------------
Test Name           State            Frequency 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
----------------------------------------------------------------------------
```

Loopback Test Statistics for the Entire Switch

The following example 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
--------------------------------------------------------------------
```

Loopback Test Statistics for a Specified Interface

The following example 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.

Loopback Test Time Log for All Modules

The following example displays 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
---------------------------------------------------------------------
```

Loopback Test Time Log for a Specified Module

The following example 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
---------------------------------------------------------------------
```

Verifying Loopback Test Configuration Frame Length

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
```

Verifying OBFL for the Switch

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
```

Verifying OBFL for 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
```

Verifying Kernel Core Collection

The kernel core collection configuration may be verified by checking the running configuration.

```
switch# show running-config | include 'kernel core'
system kernel core
```

Verifying Automatic Core Copying

Use the **show system cores** command to display the configuration of the automatic core copying feature.

```
switch# show system cores
Cores are transferred to scp://mdsadmin@192.168.1.2/tftpboot/cores
```

Displaying OBFL Logs

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

Command	Purpose
show logging onboard boot-uptime	Displays the boot and uptime information.
show logging onboard counter-stats	Displays counter statistics. **Note** In Cisco MDS 9132T and Cisco MDS 9396T switches, the output of this command displays information about removed LEM ports.
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.

Command	Purpose
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.

Displaying the Module Counters Information

This example shows the device IDs of all the devices in a module:

```
switch# attach  module 4
Attaching to module 4 ...
To exit type 'exit', to abort type '$.'
Linux lc04 2.6.10_mvl401-pc_target #1 Tue Dec 16 22:58:32 PST 2008 ppc GNU/Linux

module-4# clear asic-cnt list-all-devices
           Asic Name  |        Device ID
         Stratosphere  |               63
          transceiver  |               46
         Skyline-asic  |               57
           Skyline-ni  |               60
         Skyline-xbar  |               59
          Skyline-fwd  |               58
         Tuscany-asic  |               52
         Tuscany-xbar  |               54
          Tuscany-que  |               55
          Tuscany-fwd  |               53
       Fwd-spi-group  |               73
          Fwd-parser  |               74
                eobc  |               10
           X-Bus IO  |                1
   Power Mngmnt Epld  |               25
```

Displaying System Processes

Use the **show processes** command to obtain general information about all processes (see CPU Utilization Information, on page 27 to Memory Information About Processes , on page 29).

Displays System Processes

The following example 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:

- ProcessId = 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 = name Name of the process.

CPU Utilization Information

The following example 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:

- MemAllocated = Sum of all the dynamically allocated memory that this process has received from the system, including memory that may have been returned
- Runtime CPU Time (ms) = CPU time the process has used, expressed in milliseconds.microseconds
- 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.

Process Log Information

The following example 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.

Detail Log Information About a Process

The following example 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......*
```

All Process Log Details

The following example 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
...
```

Memory Information About Processes

The following example displays memory information about processes

```
switch# show processes memory
PID    MemAlloc  MemLimit    MemUsed     StackBase/Ptr      Process
-----  --------  ----------  ----------  -----------------  ----------------
    1    147456  0      1667072     7ffffe50/7ffff950      init
    2         0  0            0                 0/0         ksoftirqd/0
    3         0  0            0                 0/0         desched/0
    4         0  0            0                 0/0         events/0
    5         0  0            0                 0/0         khelper
```

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 Default Switch Port States, on page 29 to System Related CPU and Memory Information, on page 30).

Default Switch Port States

The following example displays default switch port states:

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

Error Information for a Specified ID

The following example displays error information for a specified ID:

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

System Reset Information

The following example 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 9700 Director, the last four reset-reason codes for the supervisor modules are displayed. If either supervisor module is absent, the reset-reason codes for that supervisor module are not displayed..
- In a Cisco MDS 9000 Series Fabric Switch, the last four reset-reason codes for the supervisor are displayed. The supervisor on a fabric switch is represented as module in slot 1.
- 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 9700 Director, this command clears the reset-reason information stored in NVRAM in the active and standby supervisor modules.
- In a Cisco MDS 9000 Series Fabric Switch, this command clears the reset-reason information stored in NVRAM in the active supervisor module.

System Uptime

The following example 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 System Related CPU and Memory Information, on page 30).

System Related CPU and Memory Information

The following example 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.

Displaying Process Fault Logs

Displaying the Process Fault Log Summary

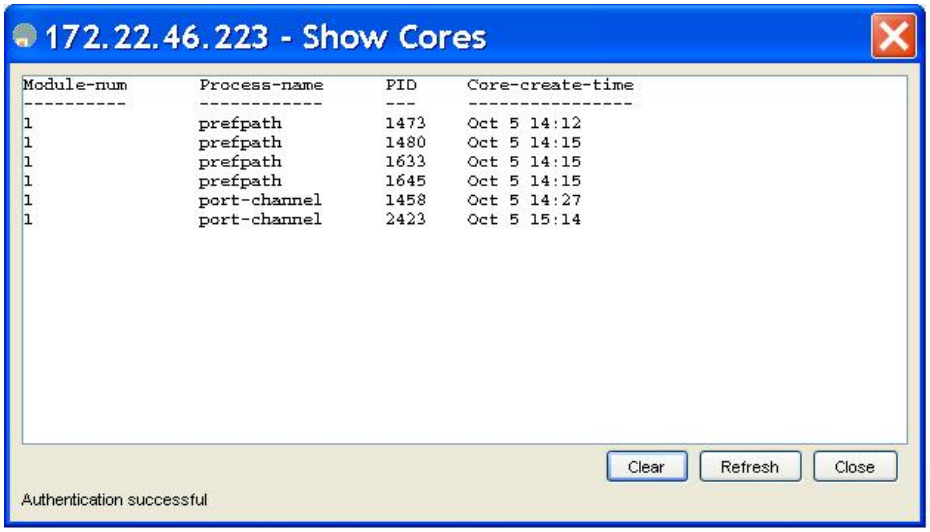
The history of fatal process faults and the logs collected of each event can be displayed on a per-module basis. Use the **slot** command to run the **show processes log** command on a particular module.

The following example displays the process fault log summary on module 2:

```
switch# slot 2 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
```

The following example displays the process cores of a system in Device Manager:

Figure 1: Show Cores Dialog Box



Displaying Process Cores

The following example displays all cores stored on 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
```

Additional References

For additional information related to implementing System Processes and Logs, see the following section:

MIBs

MIBs	MIBs Link
• CISCO-SYSTEM-EXT-MIB • CISCO-SYSTEM-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/en/US/products/ps5989/prod_technical_reference_list.html