Nexus 3500 Series Switch Platform System Health Check Process

Contents

[Introduction](#)
[Monitor CPU and Memory Usage](#)
[Check Hardware Diagnostics Status](#)
[View Hardware Profile](#)
[Active Buffer Monitoring](#)
[Monitor Interface Counters/Statistics](#)
[Monitor Control Plane Policing Statistics](#)
[Perform Bootflash File System Health Check](#)
[Collect System Cores and Process Logs](#)
[Related Information](#)

Introduction

This document describes the general process that is used in order to perform a system health check on Cisco Nexus 3500 Series switch platforms that run Nexus Operating System (NX-OS) Release 6.0(2).

Monitor CPU and Memory Usage

In order to receive an overview of the CPU and memory usage of the system, enter the **show system resources** command:

```
switch# show system resources
Load average:   1 minute: 0.32   5 minutes: 0.13   15 minutes: 0.10
Processes   :   366 total, 2 running
CPU states  :   5.5% user,   12.0% kernel,   82.5% idle
       CPU0 states  :   10.0% user,   18.0% kernel,   72.0% idle
       CPU1 states  :   1.0% user,   6.0% kernel,   93.0% idle
Memory usage:   4117064K total,   2614356K used,   1502708K free
Switch#
```

If you require more details about the processes that consume CPU cycles or memory, enter the **show process cpu sort** and **show system internal kernel memory usage** commands:

```
switch# show process cpu sort
PID    Runtime(ms)  Invoked   uSecs  1Sec    Process
-----  -----------  --------  -----  ------  -----------
 3239     55236684  24663045   2239    6.3%  mtc_usd
 3376          776      7007    110    2.7%  netstack
  15     26592500 178719270    148    0.9%  kacpid
 3441      4173060  29561656    141    0.9%  cfs
 3445      7646439   6391217   1196    0.9%  lacp
```

```
 3507     13646757  34821232    391    0.9%  hsrp_engine
   1        80564    596043    135    0.0%  init
   2            6       302     20    0.0%  kthreadd
   3         1064    110904      9    0.0%  migration/0
<snip>

switch# show system internal kernel memory usage
MemTotal:      4117064 kB
MemFree:       1490120 kB
Buffers:           332 kB
Cached:        1437168 kB
ShmFS:         1432684 kB
Allowed:       1029266 Pages
Free:           372530 Pages
Available:      375551 Pages
SwapCached:          0 kB
Active:        1355724 kB
Inactive:       925400 kB
HighTotal:     2394400 kB
HighFree:       135804 kB
LowTotal:      1722664 kB
LowFree:       1354316 kB
SwapTotal:           0 kB
SwapFree:            0 kB
Dirty:              12 kB
Writeback:           0 kB
AnonPages:      843624 kB
Mapped:         211144 kB
Slab:            98524 kB
SReclaimable:     7268 kB
SUnreclaim:      91256 kB
PageTables:      19604 kB
NFS_Unstable:        0 kB
Bounce:              0 kB
WritebackTmp:        0 kB
CommitLimit:   2058532 kB
Committed_AS: 10544480 kB
VmallocTotal:   284664 kB
VmallocUsed:    174444 kB
VmallocChunk:   108732 kB
HugePages_Total:     0
HugePages_Free:      0
HugePages_Rsvd:      0
HugePages_Surp:      0
Hugepagesize:     2048 kB
DirectMap4k:      2048 kB
DirectMap2M:   1787904 kB
switch#
```

The output shows that the **High** memory region is used by the NX-OS, and the **Low** memory region is used by the kernel. The **MemTotal** and **MemFree** values provide the total memory that is available for the switch.

In order to generate memory-usage alerts, configure the switch similar to this:

```
switch(config)# system memory-thresholds minor 50 severe 70 critical 90
```

> **Note**: For this document, the values **50**, **70**, and **90** are used only as examples; choose threshold limits based on your needs.

Check Hardware Diagnostics Status

In order to check the hardware diagnostics status, enter the **show diagnostic result all** command. Ensure that all of the tests pass, and that the **Overall Diagnostic Result** is **PASS**.

```
switch# show diagnostic result all
Current bootup diagnostic level: complete
Module 1: 48x10GE Supervisor  SerialNo : <serial #>
 Overall Diagnostic Result for Module 1 : PASS
 Diagnostic level at card bootup: complete
  Test results: (. = Pass, F = Fail, I = Incomplete, U = Untested, A = Abort)
     1) TestUSBFlash -----------------------> .
    2) TestSPROM --------------------------> .
    3) TestPCIe ---------------------------> .
    4) TestLED ----------------------------> .
    5) TestOBFL ---------------------------> .
    6) TestNVRAM --------------------------> .
    7) TestPowerSupply --------------------> .
    8) TestTemperatureSensor --------------> .
    9) TestFan ----------------------------> .
   10) TestVoltage ------------------------> .
   11) TestGPIO ---------------------------> .
   12) TestInbandPort ---------------------> .
   13) TestManagementPort -----------------> .
   14) TestMemory -------------------------> .
   15) TestForwardingEngine ---------------> .
<snip>
```

View Hardware Profile

Enter the **show hardware profile status** command in order to check the current hardware profile that is configured on the switch, and the hardware table usage:

```
switch# show hardware profile status
Hardware table usage:
Max Host Entries = 65535, Used = 341
Max Unicast LPM Entries = 24576, Used = 92
Max Multicast LPM Entries = 8192, Used (L2:L3) = 1836 (1:1835)
Switch#
```

Ensure that the usage of the **Host Entries** and **Unicast/Multicast Longest Prefix Match (LPM) Entries** are within the specified limit.

> **Note**: For optimal performance of the switch, it is important to choose the proper hardware profile template.

If you want the switch to generate a syslog at a specific threshold level, configure the switch similar to this:

```
switch(config)# hardware profile multicast syslog-threshold ?
 <1-100>  Percentage

switch(config)# hardware profile unicast syslog-threshold ?
 <1-100>  Percentage
```

> **Note**: The default threshold value is 90 percent for both unicast and multicast.

For more details, refer to the Configuring PIM Cisco article, which provides configuration details based on the license installed and features enabled. Also, if you want to optimize the forwarding table, refer to the Cisco Nexus 3000 Series Switches: Understand, Configure and Tune the Forwarding Table Cisco article.

Active Buffer Monitoring

Active Buffer Monitoring (ABM) provides the granular buffer occupancy data, which allows better insight into hot-spots of congestion. This feature supports two modes of operation: **Unicast** and **Multicast** mode.

In **Unicast** mode, ABM monitors and maintains the buffer usage data per buffer-block, and the unicast buffer utilization for all 48 ports. In **Multicast** mode, it monitors and maintains the buffer usage data per buffer-block, and the multicast buffer utilization per buffer-block.

> **Note**: For more information, reference the Cisco Nexus 3548 Active Buffer Monitoring Cisco article. Figure 4 of the article shows that the buffer usage peaked at **22:15:32** and lasted until **22:15:37**. Also, the histogram provides evidence of sudden spikes in the usage and shows the speed at which the buffer drains. If there is a slow receiver (such as a 1-Gbps receiver among 10-Gbps receivers), then in order to avoid packet drops, you must include a configuration similar to this: **hardware profile multicast slow-receiver port <x>**.

Monitor Interface Counters/Statistics

In order to monitor traffic loss, enter the **show interface ethernet x/y** command. The output from this command provides basic traffic-rate information, and also port-level drops/errors.

```
switch# show interface eth1/10
Ethernet1/10 is up
 Dedicated Interface
 Belongs to Po1
 Hardware: 100/1000/10000 Ethernet, address: 30f7.0d9c.3b51
  (bia 30f7.0d9c.3b51)
 MTU 1500 bytes, BW 10000000 Kbit, DLY 10 usec
 reliability 255/255, txload 1/255, rxload 1/255
 Encapsulation ARPA
 Port mode is trunk
 full-duplex, 10 Gb/s, media type is 10G
 Beacon is turned off
 Input flow-control is off, output flow-control is off
 Rate mode is dedicated
 Switchport monitor is off
 EtherType is 0x8100
  Last link flapped 3d21h
 Last clearing of "show interface" counters never
 14766 interface resets
 30 seconds input rate 47240 bits/sec, 68 packets/sec
  30 seconds output rate 3120720 bits/sec, 3069 packets/sec
 Load-Interval #2: 5 minute (300 seconds)
   input rate 50.18 Kbps, 52 pps; output rate 3.12 Mbps, 3.05 Kpps
```

```
  RX
   4485822 unicast packets  175312538 multicast packets  388443 broadcast
    packets
   180186040 input packets  9575683853 bytes
   0 jumbo packets  0 storm suppression bytes
   1 runts  0 giants  1 CRC  0 no buffer
   2 input error  0 short frame  0 overrun   0 underrun  0 ignored
   0 watchdog  0 bad etype drop  0 bad proto drop  0 if down drop
   0 input with dribble  260503 input discard
   0 Rx pause
  TX
   159370439 unicast packets  6366799906 multicast packets  1111 broadcast
    packets
   6526171456 output packets  828646014117 bytes
   0 jumbo packets
   0 output errors  0 collision  0 deferred  0 late collision
   0 lost carrier  0 no carrier  0 babble 0 output discard
   0 Tx pause

switch#
```

If the **input** or **output** discards show non-zero values, determine if the dropped packets are unicast and/or multicast:

```
switch# show queuing interface ethernet 1/10
Ethernet1/10 queuing information:
 TX Queuing
   qos-group  sched-type  oper-bandwidth
       0       WRR            100

 RX Queuing
    Multicast statistics:
        Mcast pkts dropped                    : 0
    Unicast statistics:
   qos-group 0
   HW MTU: 1500 (1500 configured)
   drop-type: drop, xon: 0, xoff: 0
   Statistics:
        Ucast pkts dropped                    : 0
switch#
```

The output indicates that the dropped traffic is not due to Quality of Service (QoS). Now you must check the hardware MAC address statistics:

```
switch# show hardware internal statistics device mac ?
 all         Show all stats
 congestion  Show congestion stats
 control     Show control stats
 errors      Show error stats
 lookup      Show lookup stats
 pktflow     Show packetflow stats
 qos         Show qos stats
 rates       Show packetflow stats
 snmp        Show snmp stats
```

When you perform a troubleshoot for traffic drops, the key options to check are **congestion**, **errors**, and **qos**. The **pktflow** option provides traffic statistics in the RX and TX directions, with specific packet-size ranges.

```
switch# show hardware internal statistics device mac errors port 10
|--------------------------------------------------------------------------|
```

```
| Device: L2/L3 forwarding ASIC   Role:MAC                              |
|-----------------------------------------------------------------------|
Instance:0
ID   Name                                          Value              Ports
--   ----                                          -----              -----
198  MTC_MB_CRC_ERR_CNT_PORT9                      0000000000000002   10 -
508  MTC_PP_CNT_PORT1_RCODE_CHAIN3                 0000000000000002   10 -
526  MTC_RW_EG_PORT1_EG_CLB_DROP_FCNT_CHAIN3       000000000054da5a   10 -
3616 MTC_NI515_P1_CNT_TX                           0000000000000bed   10 -
6495 TTOT_OCT                                      000000000005f341   10 -
7365 RTOT                                          0000000000000034   10 -
7366 RCRC                                          0000000000000001   10 -
7374 RUNT                                          0000000000000001   10 -
9511 ROCT                                          00000000000018b9   10 -
10678 PORT_EXCEPTION_ICBL_PKT_DROP                 000000000003f997   10 -
```

Note: The **0x3f997** hexadecimal value equals **260503** in decimal format.

```
switch# show interface eth1/10
Ethernet1/10 is up
<snip>  0 input with dribble
260503 input discard
<snip>
```

In the output, the **PORT_EXCEPTION_ICBL_PKT_DROP** error message indicates that the traffic received on the port has a **Dot1Q** tag for a VLAN that is not enabled on the switch.

Here is another example, where the traffic drop is seen due to QoS:

```
switch# show interface ethernet 1/11

Ethernet1/11 is up
<snip>
 TX

<snip>
   0 output errors  0 collision  0 deferred  0 late collision
   0 lost carrier  0 no carrier  0 babble 6153699 output discard
   0 Tx pause
switch#

switch# show queuing interface ethernet 1/11

Ethernet1/11 queuing information:
 TX Queuing
   qos-group  sched-type  oper-bandwidth
       0       WRR            100

  RX Queuing
   Multicast statistics:
       Mcast pkts dropped                    : 0
   Unicast statistics:
   qos-group 0
   HW MTU: 1500 (1500 configured)
   drop-type: drop, xon: 0, xoff: 0
   Statistics:
        Ucast pkts dropped                    : 6153699
```

> **Note**: The output indicates that **6153699** packets were dropped in the Receive-direction, which is misleading. Refer to Cisco bug ID CSCuj20713.

```
switch# show hardware internal statistics device mac all | i 11|Port

(result filtered for relevant port)
ID   Name           Value              Ports
<snip>
5596 TX_DROP        00000000005de5e3   11 -  <--- 6153699 Tx Drops in Hex
<snip>
10253 UC_DROP_VL0   00000000005de5e3   11 -  <--- Drops for QoS Group 0 in Hex
<snip>
```

In summary, here are the commands that are used in order to capture packet drops:

- **show interface ethernet x/y**
- **show queuing interface ethernet x/y**
- **show hardware internal statistics device mac errors port <port #>**

Monitor Control Plane Policing Statistics

Control Plane Policing (CoPP) protects the control plane in order to ensure network stability. For additional details, reference the Configuring Control Plane Policing Cisco article.

In order to monitor the CoPP statistics, enter the **show policy-map interface control-plane** command:

```
switch# show policy-map interface control-plane
Control Plane
 service-policy  input: copp-system-policy

   class-map copp-s-ping (match-any)
     match access-group name copp-system-acl-ping
     police pps 100 , bc 0 packets
        HW Matched Packets   30
        SW Matched Packets   30
   class-map copp-s-l3destmiss (match-any)
     police pps 100 , bc 0 packets
        HW Matched Packets   76
        SW Matched Packets   74
   class-map copp-s-glean (match-any)
     police pps 500 , bc 0 packets
        HW Matched Packets   103088
        SW Matched Packets   51544
<snip>
```

In the output, the Hardware (**HW**) and Software (**SW**) **Matched Packets** for **copp-s-ping** are the same. This means that the amount of packets that is counted by the **HW** is 30 (all sent towards the Inband CPU Driver), and the **SW** counts the same number of packets before it sends them to the CPU. This indicates that no packets are dropped by CoPP, because it is within the configured limit of 100 p/s.

When you look at the **copp-s-glean** class, which matches the packets that are destined to the IP address for which the Address Resolution Protocol (ARP) cache entry is not present, the number

of packets that is seen by the **HW** is **103,088**, while the **SW** matches only **51544**. This indicates that the CoPP dropped **51544** (103088-51544) packets, because the rate of these packets exceeds 500 p/s.

The SW counters are obtained from the CPU Inband Driver, and the HW counters come from the Access Control List (ACL) that is programmed in the HW. If you encounter a situation where the **HW Matched Packets** equal zero, and a non-zero value is present for the **SW Matched Packets**, then no ACL is present in the HW for that specific class-map, which can be normal. It is also important to note that these two counters might not be polled at the same time, and you should only use the counter values in order to troubelshoot if the difference is significant.

The CoPP statistics might not be directly related to HW-switched packets, but it is still relevant if the packets that should be sent through the switch are punted to the CPU. A packet-punt is caused by various reasons, such as when you run a glean adjacency.

Be aware that there are three types of CoPP policies: Default, Layer 2 (L2), and Layer 3 (L3). Choose the appropriate policy based on the deployment scenario, and modify the CoPP policy based on the observations. In order to fine-tune the CoPP, check regularly, and check after you obtain new services/applications or after a network redesign.

> **Note**: In order to clear the counters, enter the **clear copp statistics** command.

Perform Bootflash File System Health Check

In order to perform a health check on the bootflash file system, enter the **system health check bootflash** command:

```
switch# system health check bootflash
Unmount successful...
Checking any file system errors...Please be patient...
Result: bootflash filesystem has no errors
done.
Remounting bootflash ...done.
switch#
```

> **Caution**: The file system is unmounted when you run the test, and it is remounted once the test is complete. Ensure that the file system is not accessed while you run the test.

Collect System Cores and Process Logs

> **Caution**: Ensure that the system does not experience any process resets or crashes, and does not generate any core files or process logs when you attempt to use the commands that are mentioned in this section.

Enter these commands in order to collect the system cores and process logs:

```
switch# show cores
Module  Instance  Process-name     PID       Date(Year-Month-Day Time)
```

```
------  --------  --------------  --------  ------------------------
switch#

switch# show process log
Process          PID     Normal-exit  Stack  Core   Log-create-time
---------------  ------  -----------  -----  -----  ---------------
ethpc            4217              N      N      N  Tue Jun  4 01:57:54 2013
```

Note: Reference the Retrieving Core files from Cisco Nexus switching platforms Cisco article for more details about this process.

Related Information

- **Data Sheets and Literature - Cisco Nexus 3000 Series Switches**
- **Compare Models - Cisco Nexus 3000 Series Switches**
- **Introduction - Cisco Nexus 3000 Series Switches**
- **Understanding “Input Discard” Interface Counter in Nexus3000 - Cisco Support Communities**
- **Technical Support & Documentation - Cisco Systems**