Troubleshoot Unexpected Reloads or Crash on Nexus 9000

Contents

Introduction

This document describes how to troubleshoot unexpected reloads or crashes on Nexus 9000 switches.

Prerequisites

There are no requirements for this document.

Components Used

This document is not restricted to specific software and hardware versions.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

How the Nexus 9000 Switch Breaks

Cisco NX-OS is a resilient operating system that is specifically designed for high availability at the network, system, and process levels.

There are 3 reasons an unexpected reload can occur on Nexus 9000:

- A process in user-space could experience a crash.
- A process or piece of hardware could experience a watchdog timeout or heartbeat failure.

- The kernel itself encounters an unrecoverable condition and crashes.

Important Data to Troubleshoot Reload and Crash

- The exact date and time of the reload.
- What was happening before the reload? Any configuration changes? Any scale changes? Any logs on the device? Any environmental change? Any CPU/memory usage increase?
- Once the switch boots up and is stable, collect and check output.
- If the switch cannot up, access via the console, then check if any output. Also, check the switch LEDs. You can find LED details in the hardware installation guide.

System Reset-reason

<#root>

```
N9K#show system reset-reason module 1
```

```
----- reset reason for Supervisor-module 1 (from Supervisor in slot 1) ---
1) At 21301 usecs after Tue Jan 17 20:29:20 2023
Reason: Reset Requested due to Fatal Module Error
Service: ipfib hap reset
Version: 9.3(8)
```

Core File

<#root>

```
N9K#show cores
```

```
VDC  Module Instance  Process-name    PID      Date(Year-Month-Day Time)
--- ------ -------- -------------- -------- ------------------------
```

```
A     B       C         D             E          2024-01-04 19:17:25
copy core://<module-number>/<process-id>[/instance-num]
copy core://B/E/C ftp://<address>/<directory>
```

Onboard Logs

<#root>

show logging onboard

show logging onboard kernel-trace

show logging onboard stack-trace

```
************************************************************
STACK TRACE GENERATED AT Sun Sep 10 19:06:39 2023 CCT
************************************************************
<snip>                                                              >>>dumps kernel massages
<0>[10925084.972289] [1694343998] sysServices Unexpected call in interrupt context, serviceId=824
<0>[10925084.980666] [1694343998] cctrl_set_card_offline - EOBC switch reset failed
<0>[10925084.987824] [1694343998] sysServices Unexpected call in interrupt context, serviceId=824
<0>[10925084.996200] [1694343998] cctrl_set_card_offline - EPC switch reset failed
<snip>
<4>[10925085.040600] [1694343998] Dumping interrupt statistics                >>>dump interrupt statict
<4>[10925085.045928] [1694343998] CPU0 CPU1
<4>[10925085.051732] [1694343998] 3: 0 0 axp_irq Armada Error Handler
<4>[10925085.059909] [1694343998] 4: 0 0 axp_irq Armada MBUS unit Error Handle
<4>[10925085.068957] [1694343998] 5: 1012335907 809985523 axp_irq axp_local_clockevent
<4>[10925085.077136] [1694343998] 8: 1260801154 0 axp_irq mv_eth
<4>[10925085.084108] [1694343998] 31: 11230 0 axp_irq mv64xxx_i2c
<4>[10925085.091508] [1694343998] 41: 7111 1 axp_irq serial
<4>[10925085.098471] [1694343998] 51: 2 0 axp_irq mv_xor.0
<4>[10925085.105602] [1694343998] 52: 2 0 axp_irq mv_xor.1
<4>[10925085.112760] [1694343998] 94: 1 0 axp_irq mv_xor.2
<4>[10925085.119890] [1694343998] 95: 1 0 axp_irq mv_xor.3
<4>[10925085.127029] [1694343998] 107: 0 0 axp_irq axp-temp
<4>[10925085.134200] [1694343998] 168: 0 0 axp_irq cctrl_mrv_nmi_irq
<4>[10925085.142134] [1694343998] 195: 29 0 axp_msi_irq cctrl_sc_msi_irq
<4>[10925085.150225] [1694343998] 196: 0 2399172865 axp_msi_irq linux-kernel-bde
<4>[10925085.158325] [1694343998] IPI0 : 0 0 Timer broadcast interrupts
<4>[10925085.166130] [1694343998] IPI1 : 1711470501 3532640372 Rescheduling interrupts
<4>[10925085.173672] [1694343998] IPI2 : 0 0 Function call interrupts
<4>[10925085.181302] [1694343998] IPI3 : 44582 118572 Single function call interrupts
<4>[10925085.189541] [1694343998] IPI4 : 0 0 CPU stop interrupts
<4>[10925085.196734] [1694343998] PMU: : 0 0
<4>[10925085.202186] [1694343998] Err : 0
```

show logging onboard exception-log

```
    >>>Check if any exception is raised before reload
```

Process Log

<#root>

```
N9K# show processes log details

                                    >>>detail process memory usage prior to crash
Service: ethpm
Description: Test Ethernet Port Manager
Executable: /isan/bin/ethpm

Started at Wed Jun 5 18:20:46 2023 (251615 us)
Stopped at Sat Jun 8 00:08:53 2023 (661042 us)
Uptime: 2 days 5 hours 48 minutes 7 seconds

Start type: SRV_OPTION_RESTART_STATELESS (23)
Death reason: SYSMGR_DEATH_REASON_FAILURE_SIGNAL (2)
Last heartbeat 48.10 secs ago
System image name:
System image version: 7.0(3)I7(6)
PID: 28914
Exit code: signal 5 (core dumped)
CWD: /var/sysmgr/work

RLIMIT_AS: 1019819820

                                              >>>limit memory usage
Virtual Memory:
CODE 1007E000 - 1068DBD4
DATA 1068E000 - 106DC3E8
BRK 1194F000 - 11CF9000
STACK FFA28650

TOTAL 576004 KB

                                                    >>>memory usage before crash
```

Logfiles from Logflash

There is a build-in logflash on Nexus 9000, log files survive after reloading.

<#root>

```
N9K#dir logflash:log | grep messages


3714961 Jan 13 18:05:31 2024 messages
4194331 Jan 13 17:30:14 2021 messages.1
5497842 May 11 15:59:00 2021 messages.2
4194341 Jul 30 07:25:36 2022 messages.3
4194510 Feb 09 14:50:50 2023 messages.4
4194426 Jun 04 05:00:40 2023 messages.5

N9K#show file logflash:log/messages
N9K#show file logflash:log/messages.1
N9K#show file logflash:log/messages.2
```

```
N9K#show file logflash:log/messages.3
N9K#show file logflash:log/messages.4
N9K#show file logflash:log/messages.5
```

Common Reset-Reasons

Power Related Reload

<#root>

N9K#show system reset-reason

```
----- reset reason for module 1 (from Supervisor in slot 1) ---
1) At 280125 usecs after Fri Aug 4 02:01:14 2023
```

Reason: Module PowerCycled

```
Service: HW check by card-client
Version:
```

Explanation

Nexus 9000 switch supports N+1 power redundancy. If a power outage happens on most or all power sources, a reload occurs.

Recommended:

1. Verify the power cords of the power supplies.

2. Check if other devices sharing the same inlet circuit also had an outage.

3. Check if any power-related alarm on Nexus 9000 or PDU.

Process Crash

<#root>

N9K#show system reset-reason module 1

```
----- reset reason for Supervisor-module 1 (from Supervisor in slot 1)
1) At 21301 usecs after Tue Jan 17 20:29:20 2023
Reason: Reset Requested due to Fatal Module Error
```

Service: ipfib hap reset

```
                                          >>>ipfib process reset
Version: 9.3(8)
```

Explanation

Each service has its own High Availablity (HA) policy, including a heartbeat timer, restart method, and stateful restart max retry. Cisco NX-OS Software allows stateful restarts of most processes and services. The reload occurs if the process ha policy is reset (NX-OS cannot work during process restart) or the times of process restart reach max retry.

Recommended

<#root>

`show cores`

```
VDC  Module  Instance  Process-name     PID       Date(Year-Month-Day Time)
---  ------  --------  ---------------  --------  -------------------------
1    1       1         ipfib            27446     2023-01-17 20:30:30
copy core://1/27446/1 ftp://<address>/<directory>
```

Most of the process crash is software defect and the core file is saved, open a service request case to confirm.

- Core files can be decoded by the TAC engineer.
- To open the service request please choose **Product** > **Unexpected Reboot** > **Software Failure** to get the case opened with the right team.

EOBC Failure

```
2018 Jan 21 01:56:42.789 N9K#%KERN-0-SYSTEM_MSG: [4590707.849157] [1516460202] EMON: module 2 is not re
2018 Jan 21 01:56:43.071 N9K#%MODULE-2-MOD_DIAG_FAIL: Module 2 (Serial number: xxxxxxxxxx) reported fai
```

Explanation

The EOBC is short for Ethernet Out of Band Channel. Regular keepalives are going between the supervisor and line cards. The error messages you received indicate a heartbeat went missing between SUP and linecard. If a single heartbeat goes missing, it can be ignored automatically. However, if multiple heartbeats are lost simultaneously, then the line card would be reset.

There are usually 3 reasons for EOBC failure:
1. EOBC congestion. You can see more than 1 linecard experience EOBC lost.
2. CPU hog in specific module(s). Linecard/supervisor CPU is busy and not able to handle EOBC messages. There is a software enhancement starting from Nexus 9000 from 7.0(3)I7(3).
3. Hardware fault.

Recommended

1. Check if any CPUhog for impacted linecard around reload.

2. Check if other linecard experience EOBC loss around reload.

3. Check if deployed BFD or Netflow CPU consuming service recently.

4. If it occurs multiple times without any information, replace the hardware.

Parity Error

<#root>

```
N9K#show logging onboard stack-trace

************************************************************
        STACK TRACE GENERATED AT Tue Sep 21 02:27:58 2021 UTC
************************************************************
<0>[88302546.800770] [1632158876] ERROR: MACHINE: Uncorrectable
<0>[88302546.809202] [1632158876] L2CACHE ERROR: Cause 0x88

<0>[88302546.814368] [1632158876] TAG Parity Error

                           >>>>>Parity error
<0>[88302546.818750] [1632158876] Kernel panic - not syncing: L2CACHE ERROR
<4>[88302546.825212] [1632158876] Cpu: 0 Pid: 0, comm:            swapper/0
```

Explanation

A parity error occurs when a bit of information is flipped from 1 to 0 or 0 to 1.

Most of the parity errors are caused by electrostatic or magnetic-related environmental conditions. These events randomly occur and are not able to be prevented.

Systems detect that this error has occurred and force the system to crash to prevent incorrect data from being processed. One occurrence is not an indication of a hardware or software problem.

Recommended

Parity errors can be transient single-event upsets (SEU), or they can be caused by defective hardware. To determine which this is, you need to monitor the device for 48 hours to see if it has a recurrence.

If no second occurrence within 48 hours, the issue is considered transient, no action is needed.

Frequent or repeatable (hard) parity errors are caused by physical malfunction of the memory or the circuitry used to read and write. In such cases, replace the hardware.

PCIE Error

<#root>

```
N9K#show logging onboard stack-trace


<6>[  105.196227]     CCTRL PANIC DUMP
<6>[  105.196229] ========================
<6>[  105.196231] WDT last punched at 105192052644
<6>[  105.196234] REG(0x60) = 3c
<6>[  105.196238] REG(0x64) = 0
```

```
<6>[  105.196241] REG(0x300) = baadbeef
<6>[  105.196245] REG(0x304) = baadbeef
<6>[  105.196246] ========================
```

<0>[105.197303] nxos_panic: Kernel panic - not syncing: PCIE Uncorrectable error

```
   >>>>>PCIE Uncorrectable error
```

Explanation

PCIE errors are classified into two types: correctable errors and uncorrectable errors. This classification is based on the impact of those errors, which results in degraded performance or function failure.

Correctable errors pose no impact on the functionality of the interface. The PCIE protocol can recover without any software intervention or any loss of data. These errors are detected and corrected by hardware.

Uncorrectable errors impact the functionality of the interface. Uncorrectable errors can cause a particular transaction or a particular PCIE link to be unreliable. Depending on those error conditions, uncorrectable errors are further classified into non-fatal errors and fatal errors. Non-fatal errors cause the particular transaction to be unreliable, but the PCIE link itself is fully functional. Fatal errors, on the other hand, cause the link to be unreliable.

Nexus 9000 detects fatal PCIE errors and forces the system to reload to prevent incorrect data from being processed.

Recommended

Same with parity error.

If no second occurrence within 48 hours, the issue is considered transient, no action is needed.

Frequent or repeatable errors are caused by physical malfunction. In such cases, replace the hardware.

Watchdog Timeout

<#root>

N9K#show system reset-reason

```
----- reset reason for module 1 (from Supervisor in slot 1) ---
1) At 88659 usecs after Mon Sep 24 18:33:04 2023
```

**   Reason: Watchdog Timeout**

```
   Service:
   Version: 7.0(3)I7(9)
```

Explanation

Watchdog timers are commonly found in embedded systems and other computer-controlled equipment where humans cannot easily access the equipment or would be unable to react to faults in a timely manner. Nexus 9000 deploys a watchdog timer feature via FPGA. This ensures Nexus 9000 can detect software hang and reboot the switch promptly.

Recommended

1. Verify if any known software bugs impact the current version.

2. If the issue re-occurs, collect kernel trace and any additional logging data.

3. Open a service request case.

Manually Reload due to CLI or Upgrade

<#root>

```
N9K# show system reset-reason


----- reset reason for module 1 (from Supervisor in slot 1) ---
1) At 343832 usecs after Sat

Jan 13 17:58:53 2024


Reason: Reset Requested by CLI command reload


Service:
Version: 10.2(5)
>
4) At 282886 usecs after Fri

Jan 12 07:42:33 2024


Reason: Reset due to upgrade


Service:

Version: 10.3(4a)

                >>>>>version prior to upgrading
```

Explanation

The Nexus 9000 Series switches support disruptive software upgrades and downgrades by default. Nexus 9000 reloads during the upgrade.

Recommended

Expected behavior. Check the accounting log for more CLI session details.

CLI reload example:

```
Sat Jan 13 17:58:40 2024:type=update:id=console0:user=admin:cmd=reload (REDIRECT)
Sat Jan 13 17:58:47 2024:type=update:id=console0:user=admin:cmd=Rebooting the switch
```

Upgrade reload example:

```
Fri Jan 12 07:35:52 2024:type=update:id=console0:user=admin:cmd=install all nxos bootflash:/nxos64-cs.1
```

Cisco Bug IDs

Some of the defects can cause an unexpected reload on Nexus 9000 switches. To confirm if you hit a known software bug, please open a TAC case.

Cisco bug ID	Bug title	Fix version
Cisco bug ID CSCwd53591	Reload due to watchdog timeout without cores/traces	9.3(13)
Cisco bug ID CSCvz65993	tahoe0 brought down resulting in inband connectivity failure	9.3(9)
Cisco bug ID CSCvs00400	Kernel panic and reload due to Watchdog Timeout after link flaps	9.3(3) and 7.0(3)I7(8)
Cisco bug ID CSCvr57551	Cisco Nexus 9000 reloads with Kernel panic - unable to handle kernel paging request	7.0(3)I7(8) and 9.3(4)
Cisco bug ID CSCvo86286	Kernel panic seen on 7.0(3)I7(x) with Nexus 9500 1st Gen line cards	7.0(3)I7(7)
Cisco bug ID CSCvx38752	Memory leak causing Nexus 9k to reload "ipfib"	7.0(3)I7(9) and 9.3(2)
Cisco bug ID CSCvh13039	LC/FM reloads due to EOBC heartbeat as CPU busy servicing hrtimer	7.0(3)I4(8) and 7.0(3)I7(3)