Identify and Trace Nexus 9000 Cloud Scale ASIC CRC

Contents

- Introduction
- Prerequisites
  - Requirements
  - Components Used
- Background Information
  - Applicable Hardware
    - Nexus 9200/9300 Fixed Switches
    - Nexus 9500 Modular Switch Line Cards
- Cisco Nexus 9200 and 9300 Cloud Scale CRC Identification and Tracing Procedure
  - NX-OS Software Release 10.2(1) and Later
  - NX-OS Software Release 10.1(2) and Earlier
    - Step 1. Identify Incrementing CRC Counters on Physical Interface(s)
    - Step 2. Map the Physical Interface to ASIC, MAC Block, and Mac Block Sub-Port
    - Step 3. Check Cloud Scale ASIC Registers for CRC-Related Counters
- Cisco Nexus 9500 Cloud Scale - CRC Identification and Tracing Procedure on Modular Switches
  - Step 1. Map Internal Links Between Line Cards and Fabric Modules.
  - Step 2. Check the CRC counters on iEth Links and Track the Source of Corrupted Frames
- Examples
  - Scenario 1. Physical Interface Receiving Stomped CRCs
    - Step 1. Confirm Incrementing CRCs
    - Step 2. Map Physical Interface to ASIC, MAC Block, and MAC Block Sub-Port
    - Step 3. Check Cloud Scale ASIC Registers for CRC-Related Counters
    - Conclusion
  - Scenario 2. Physical Interface Received Malformed Frames with Invalid CRC
    - Step 1. Confirm Incrementing CRCs
    - Step 2. Map Physical Interface to ASIC, MAC Block, and MAC Block Sub-Port
    - Step 3. Check Cloud Scale ASIC Registers for CRC-Related Counters
    - Conclusion
  - Scenario 3. Nexus 9500 iEth CRC Errors Syslog
    - Step 1. Map iEth Link on Fabric Module to Connected Line Card
    - Step 2. Check if CRCs Received on iEth Link are Invalid or Stomped
    - Step 3. Track Source of Frames with Invalid CRCs on Ingress Line Card
    - Conclusion
  - Scenario 4. Track Source of Invalid CRC Frames with Egress Interface
    - Step 1. Identify Fabric Module Sending Invalid CRC Frames to Egress Line Card
    - Step 2. Map iEth Link on Fabric Module to the Connected Linecard and Check for Stomped CRCs
    - Step 3. Track Source of Frames with Invalid CRCs on Ingress Module
    - Conclusion
- Related Information

Introduction

This document describes the steps used to trace the source of CRC errors observed on Cisco Nexus 9000 Cloud Scale ASIC modules.

Prerequisites

Requirements

Cisco recommends that you understand the basics of cut-through and store-and-forward switching. Cisco also recommends that you understand the basics of the Ethernet Frame Check Sequence (FCS) field and the Cyclic Redundancy Check (CRC) algorithm used by the FCS field. For more information, refer to this document:

- Cut-Through and Store-and-Forward Ethernet Switching for Low-Latency Environments

Components Used

The information in this document is based on Cisco Nexus 9000 series switches with the Cloud Scale ASIC running NX-OS software release 7.0(3)I7(8).

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.

Background Information

This document also describes the procedure used to differentiate stomped and non-stomped CRC errors observed on physical interfaces and internal fabric links of modular Nexus switches.

Cisco Nexus 9000 series switches use cut-through switching by default. Cut-through switching is where a switch makes a forwarding decision on a frame and begins forwarding the frame out of an egress interface as soon as the switch has processed enough of the frame's header to make a valid forwarding decision. This differs from store-and-forward switching, where a switch buffers the entire frame prior to forwarding the frame out of an egress interface.

The FCS field of an Ethernet frame validates the integrity of the frame and ensures that the frame has not been corrupted in transit. The FCS field of an Ethernet frame is located at the end of the Ethernet frame behind the frame's payload. A switch operating in a store-and-forward switching mode is able to verify the integrity of an Ethernet frame with the FCS field prior to forwarding the frame out of an egress interface (or drop the frame if the FCS field has invalid contents). However, a switch operating in a cut-through switching mode is not able to verify the integrity of an Ethernet frame with the FCS field prior to forwarding the frame out of an egress interface; in other words, by the time a cut-through switch is able to verify the integrity of an Ethernet frame, the majority of the Ethernet frame has already been forwarded out of an egress interface.

If a switch operating in a cut-through switching mode receives an Ethernet frame with an invalid FCS field, the switch can take these actions:

1. Calculate the CRC value of the incoming corrupted frame, then rewrite the FCS field of the Ethernet frame with the bitwise inverse of the calculated CRC value. If the frame needs to be routed, the CRC value of the corrupted frame would be calculated and bitwise inversed after the frame's Ethernet

header is rewritten. This action is known as "stomping" the CRC.
2. Forward the remainder of the Ethernet frame (along with the stomped CRC) out of the egress interface according to the forwarding decision made on the frame.
3. Increment the input errors counter and/or CRC errors counter on the ingress interface.

This document describes the steps to verify whether CRC counters associated with an ingress interface are normal CRCs (which typically indicate physical layer issues on the link connected to the ingress interface) or stomped CRCs (which indicate that the device connected to the ingress interface is also operating in a cut-through switching mode and received a malformed Ethernet frame).

Applicable Hardware

The procedure covered in this document is applicable to this hardware only:

Nexus 9200/9300 Fixed Switches

- N9K-C92160YC-X
- N9K-C92300YC
- N9K-C92304QC
- N9K-C92348GC-X
- N9K-C9236C
- N9K-C9272Q
- N9K-C9332C
- N9K-C9364C
- N9K-C93108TC-EX
- N9K-C93108TC-EX-24
- N9K-C93180LC-EX
- N9K-C93180YC-EX
- N9K-C93180YC-EX-24
- N9K-C93108TC-FX
- N9K-C93108TC-FX-24
- N9K-C93180YC-FX
- N9K-C93180YC-FX-24
- N9K-C9348GC-FXP
- N9K-C93240YC-FX2
- N9K-C93216TC-FX2
- N9K-C9336C-FX2
- N9K-C9336C-FX2-E
- N9K-C93360YC-FX2
- N9K-C93180YC-FX3
- N9K-C93108TC-FX3P
- N9K-C93180YC-FX3S
- N9K-C9316D-GX
- N9K-C93600CD-GX
- N9K-C9364C-GX
- N9K-C9364D-GX2A
- N9K-C9332D-GX2B

Nexus 9500 Modular Switch Line Cards

- N9K-X97160YC-EX
- N9K-X9732C-EX
- N9K-X9736C-EX

- N9K-X97284YC-FX
- N9K-X9732C-FX
- N9K-X9788TC-FX
- N9K-X9716D-GX

Cisco Nexus 9200 and 9300 Cloud Scale CRC Identification and Tracing Procedure

This section of the document describes step-by-step instructions to identify the source of CRC errors observed on a specific physical interface Ethernet1/1 on Cisco Nexus 9200 and 9300 series switches.

NX-OS Software Release 10.2(1) and Later

Starting with NX-OS software release 10.2(1), Nexus switches equipped with the Cloud Scale ASIC have a new interface counter for packets with a stomped CRC in the FCS field of Ethernet frames that traverse the switch. You can use the **show interface** command to identify physical interfaces with incrementing non-zero CRC and stomped CRC counters. An example of this is shown here, where physical interface Ethernet1/1 has a zero CRC counter and non-zero stomped CRC counter, which indicates that frames with an invalid and stomped CRC were received on this interface.

<#root>

switch#

show interface

```
<snip>
Ethernet1/1 is up
admin state is up, Dedicated Interface
Hardware: 100/1000/10000/25000 Ethernet, address: 00d7.8f86.2bbe (bia 00d7.8f86.2bbe)
MTU 1500 bytes, BW 10000000 Kbit, DLY 10 usec
reliability 255/255, txload 1/255, rxload 1/255
Encapsulation ARPA, medium is broadcast
Port mode is trunk
full-duplex, 10 Gb/s, media type is 10G
Beacon is turned off
Auto-Negotiation is turned on FEC mode is Auto
Input flow-control is off, output flow-control is off
Auto-mdix is turned off
Rate mode is dedicated
Switchport monitor is off
EtherType is 0x8100
EEE (efficient-ethernet) : n/a
admin fec state is auto, oper fec state is off
Last link flapped 04:09:21
Last clearing of "show interface" counters 00:50:37
0 interface resets
  RX
    8 unicast packets  253 multicast packets  2 broadcast packets
    1832838280 input packets  2199405650587 bytes
    0 jumbo packets  0 storm suppression bytes
    0 runts  0 giants
```

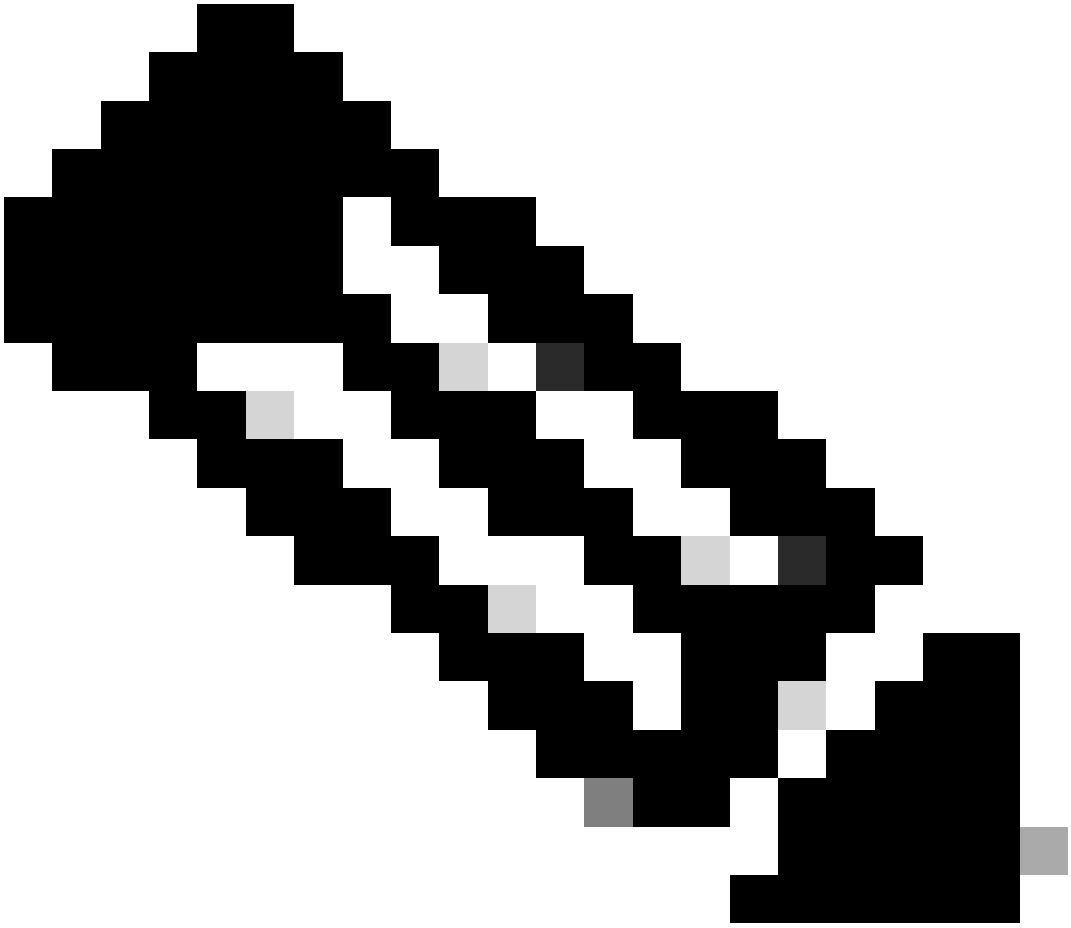
1832838019

CRC

```
  0 no buffer
    1832838019 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  0 input discard
    0 Rx pause
```

1832838019 Stomped CRC

```
  TX
    908 unicast packets  323 multicast packets  3 broadcast packets
    1234 output packets  113342 bytes
    0 jumbo packets
    0 output error  0 collision  0 deferred  0 late collision
    0 lost carrier  0 no carrier  0 babble  0 output discard
    0 Tx pause
```



Note: An incrementing CRC counter indicates a frame was received with either stomped CRC or an invalid, but non-stomped CRC. An incrementing stomped CRC counter increase indicates a frame with a stomped CRC was received.

Alternatively, the **show interface counters errors non-zero** command can be used to see interface errors counters. An example of this is shown here.

```
<#root>

switch#

show interface counters errors non-zero


-------------------------------------------------------------------------------
Port          Align-Err    FCS-Err   Xmit-Err

Rcv-Err

  UnderSize OutDiscards
-------------------------------------------------------------------------------
Eth1/1

1790348828


1790348828

         0

1790348828

         0           0

-------------------------------------------------------------------------------
Port         Single-Col  Multi-Col   Late-Col  Exces-Col  Carri-Sen       Runts
-------------------------------------------------------------------------------

-------------------------------------------------------------------------------
Port          Giants SQETest-Err Deferred-Tx IntMacTx-Er IntMacRx-Er Symbol-Err
-------------------------------------------------------------------------------

-------------------------------------------------------------------------------
Port         InDiscards
-------------------------------------------------------------------------------

-------------------------------------------------------------------------------
Port

Stomped-CRC


-------------------------------------------------------------------------------
Eth1/1

1790348828
```

You can pipe the **show interface** command to the **json** or **json-pretty** commands to obtain CRC and stomped CRC counter statistics in a structured format. An example of this is shown here.

```
<#root>

switch#
```

```
show interface Ethernet1/1 | json-pretty | include ignore-case crc


            "eth_crc": "828640831",
            "

eth_stomped_crc

": "

828640831

",
```

The NX-API REST API can be used to retrieve these same statistics using the sys/intf/phys-[intf-id]/dbgEtherStats.json object model. An example of this is shown here.

<#root>

```
/api/node/mo/sys/intf/phys-[eth1/1]/dbgEtherStats.json


{
  "totalCount": "1",
  "imdata": [
    {
      "rmonEtherStats": {
        "attributes": {
          "cRCAlignErrors": "26874272810",
          "dn": "sys/intf/phys-[eth1/1]/dbgEtherStats",
          "dropEvents": "0",
          "rXNoErrors": "26874276337",

          "stompedCRCAlignErrors":

"26874272810"

,


     ...
        }
      }
    }
  ]
}
```

NX-OS Software Release 10.1(2) and Earlier

For NX-OS software releases prior to 10.2(1), the stomped CRC counter is not available on interfaces. Several steps are needed to determine the ingress interface where invalid CRCs are observed and validate whether the CRCs are invalid or stomped.

Step 1. Identify Incrementing CRC Counters on Physical Interface(s)

Use the **show interface** command to identify physical interfaces with incrementing non-zero CRC counters. An example of this shown here, where physical interface Ethernet1/1 has a non-zero CRC counter.

```
<#root>

switch#

show interface


<snip>
Ethernet1/1 is up
admin state is up, Dedicated Interface
  Hardware: 100/1000/10000/25000 Ethernet, address: 00d7.8f86.2bbe (bia 00d7.8f86.2bbe)
  MTU 1500 bytes, BW 10000000 Kbit, DLY 10 usec
  reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, medium is broadcast
  Port mode is trunk
  full-duplex, 10 Gb/s, media type is 10G
  Beacon is turned off
  Auto-Negotiation is turned on  FEC mode is Auto
  Input flow-control is off, output flow-control is off
  Auto-mdix is turned off
  Rate mode is dedicated
  Switchport monitor is off
  EtherType is 0x8100
  EEE (efficient-ethernet) : n/a
    admin fec state is auto, oper fec state is off
  Last link flapped 04:09:21
  Last clearing of "show interface" counters 00:50:37
  0 interface resets
  RX
    3 unicast packets  3087 multicast packets  0 broadcast packets
    3097 input packets  244636 bytes
    7 jumbo packets  0 storm suppression bytes
    0 runts  7 giants

7 CRC

  0 no buffer
    7 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  0 input discard
    0 Rx pause
```

Alternatively, you can use the **show interface counters errors non-zero** command to display all interface with non-zero error counters (which includes non-zero CRC counters). An example of this shown here, where physical interface Ethernet1/1 has a non-zero CRC counter displayed by the FCS-Err column.

```
<#root>

switch#

show interface counters errors non-zero


<snip>
--------------------------------------------------------------------------------

Port

          Align-Err

FCS-Err
```

```
   Xmit-Err    Rcv-Err  UnderSize OutDiscards
------------------------------------------------------------------------------
Eth1/1
                7
7
          0          7          0          0
```

Step 2. Map the Physical Interface to ASIC, MAC Block, and Mac Block Sub-Port

Use the **show interface hardware-mappings** command to identify three key characteristics:

1. Unit - The identifier of the Cloud Scale ASIC that the physical interface connects to. This uses a zero-based numbering system (for example, the first ASIC is 0, the second ASIC is 1, and so on)
2. MacId - The identifier of the MAC block that the physical interface connects to. This uses a zero-based numbering system (for example, the first MAC block is 0, the second MAC block is 1, and so on)
3. MacSP - The identifier of the MAC block sub-port that the physical interface connects to. Each MAC block has four sub-ports associated with it, which use a zero-based numbering system and increment by a value of 2. Therefore, the first sub-port has an index of 0, the second sub-port has an index of 2, the third sub-port has an index of 4, and the fourth sub-port has an index of 6.

This is demonstrated in the example here, where physical interface Ethernet1/1 is associated with Cloud Scale ASIC 0, MAC block 4, and MAC block sub-port 0.

```
<#root>

switch#
show interface hardware-mappings

<snip>
------------------------------------------------------------------------------------------------------------------
Name       Ifindex  Smod
Unit
 HPort FPort NPort VPort Slice SPort SrcId
MacId MacSP
 VIF  Block BlkSrcID
------------------------------------------------------------------------------------------------------------------
Eth1/1
     1a000000 1
0
    16    255   0     -1    0     16    32
4     0
     1    0     32
Eth1/2     1a000200 1    0    17    255   4     -1    0     17    34    4     2     5    0     34
Eth1/3     1a000400 1    0    18    255   8     -1    0     18    36    4     4     9    0     36
```

```
Eth1/4      1a000600 1    0    19    255   12    -1    0     19    38    4     6     13    0     38
Eth1/5      1a000800 1    0    12    255   16    -1    0     12    24    3     0     17    0     24
```

Step 3. Check Cloud Scale ASIC Registers for CRC-Related Counters

Use the **slot {x} show hardware internal tah counters asic {y}** command to view register counters for the Cloud Scale ASIC. This command contains two variables:

1. {x} - Replace this value with the line card slot number. For top-of-rack switches, this is always a value of 1. For end-of-row modular switches, the line card slot number is the first number in the physical interface's name. For example, physical interface Ethernet1/1 would have a line card slot number of 1, while physical interface Ethernet4/24 would have a line card slot number of 4.
2. {y} - Replace this value with the Cloud Scale ASIC identifier identified in Step 2. For example, if the Unit column for physical interface Ethernet1/1 had a value of 0, then the value of this variable would be 0. If the Unit column for physical interface Ethernet4/24 had a value of 3, then the value of this variable would be 3.

This output can display a table. Each row of the table is a different ASIC register. Each column of the table corresponds with a physical interface on the switch. The name used for each column is not the name of the physical interface, but is a combination of the MAC block and MAC block sub-port. The format used for the column header is this:

<#root>

M

{A}

,

{B}

-

{InterfaceSpeed}

There are three variables in this format, which are:

1. {A} - Replace this value with the MAC block number.
2. {B} - Replace this value with the MAC block sub-port number.
3. {InterfaceSpeed} - This value can correspond with the physical speed of the interface (for example, 10G, 25G, 40Gx4, and so on)

This is demonstrated in the example here. Recall that physical interface Ethernet1/1 is associated with line card slot number 1 and Cloud Scale ASIC 0, which means the command you must run is **slot 1 show hardware internal tah counters asic 0**. The MAC block associated with physical interface Ethernet1/1 is 4, the MAC block sub-port associated with physical interface Ethernet1/1 is 0, and physical interface Ethernet1/1 is a 10G interface. Therefore, the column header we are looking for is M4,0-10G.

Note: The output of this command is very long and wide. It can be difficult to read this output within a terminal session. Cisco recommends maximizing the width of your terminal with the **terminal width 511** command and copying this output to an external text reader/editor for review.

```
<#root>

switch#

slot 1 show hardware internal tah counters asic 0

<snip>
*************** PER MAC/CH SRAM COUNTERS ****************
REG_NAME

M4,0-10G

     M4,2-10G     M4,4-10G     M4,6-10G     M5,0-40Gx4   M6,0-40Gx4   M7,0-40Gx4   M8,0-10G
-------------------------------------------------------------------------------------------------
02-RX Frm with FCS Err   ....         ....         ....         ....         ....         ....
16-RX Frm CRC Err(Stomp) c            ....         ....         ....         ....         ....
```

The output of this command contains several dozen register counters. There are two key register counters that are related to differentiating natural CRC errors from stomped CRCs:

1. **02-RX Frm with FCS Err** - Indicates a frame with an invalid, but non-stomped CRC was received.
2. **16-RX Frm CRC Err(Stomp)** - Indicates a frame with a stomped CRC was received.

The value of these counters is in hexadecimal. The dec NX-OS command can convert a hexadecimal value to a decimal value, as shown here.

```
<#root>

N9K-C93180YC-EX-2#

dec 0xc

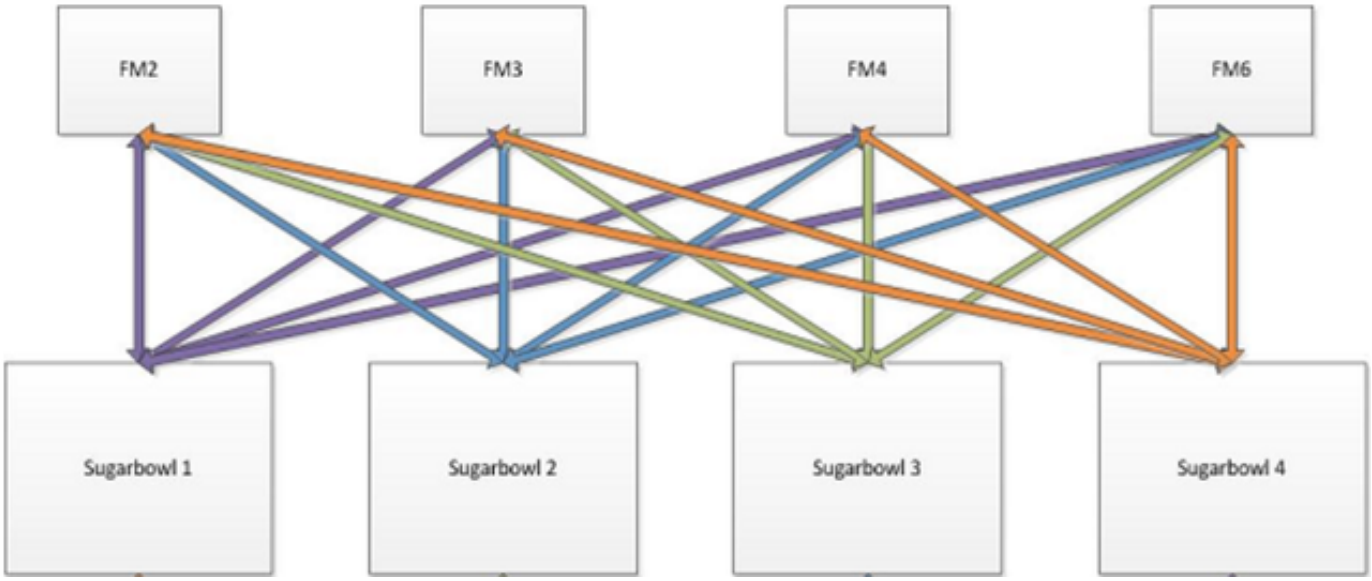

12
```

The combined values of both register counters is equivalent to the number of CRCs observed on the physical interface through the output of **show interface** or **show interface counters errors non-zero**.

Cisco Nexus 9500 Cloud Scale - CRC Identification and Tracing Procedure on Modular Switches

This section of the document describes step-by-step instructions to identify the source of CRC errors observed on a specific physical interface Ethernet1/1 on Cisco Nexus 9500 series switches.

Each line card on a Nexus 9500 series switch is connected via internal link (iEth) to the fabric modules. Each ASIC of each line card has full-mesh connectivity to all fabric modules. The example here shows a line card with four Sugarbowl ASICs with internal links connecting to four fabric modules within a modular Nexus 9500 switch.



When traffic received by an ASIC needs to egress another ASIC or line card, that traffic needs to be sent to the fabric through a fabric module. The ingress ASIC selects one of the iEth links to the fabric modules based on a hash of the packet's headers and the number of iEth links available to the ASIC.

Step 1. Map Internal Links Between Line Cards and Fabric Modules.

Use the **show system internal fabric connectivity module {x}** command (where {x} is the line card or fabric module slot number) to display the internal links between the specified line card and all fabric modules. This output displays a table where each row shows a one-to-one mapping between line card internal links (under the LC-iEthLink column) to each fabric module's internal links (under the FM-iEthLink column). An example of this is shown here, taken from a Nexus 9508 switch with 8 line cards and 4 fabric modules inserted. The output here shows that each ASIC instance of the line card inserted in slot 8 of the switch is connected to each of the 4 installed fabric modules (inserted in slots 22, 23, 24, and 26) through 2 internal links.

<#root>

```
Nexus9500#
```

```
show system internal fabric connectivity module 8
```

```
Internal Link-info Linecard slot:8
------------------------------------------------------------------------
 LC-Slot  LC-Unit   LC-iEthLink   MUX   FM-Slot   FM-Unit   FM-iEthLink
------------------------------------------------------------------------
    8        0     iEth01        -        22        0      iEth18
    8        0     iEth02        -        22        1      iEth50
    8        0     iEth03        -        23        0      iEth18
    8        0     iEth04        -        23        1      iEth50
    8        0     iEth05        -        24        0      iEth18
    8        0     iEth06        -        24        1      iEth50
    8        0     iEth07        -        26        0      iEth18
    8        0     iEth08        -        26        1      iEth50
    8        1     iEth09        -        22        0      iEth03
    8        1     iEth10        -        22        1      iEth35
    8        1     iEth11        -        23        0      iEth03
    8        1     iEth12        -        23        1      iEth35
    8        1     iEth13        -        24        0      iEth03
    8        1     iEth14        -        24        1      iEth35
```

```
    8        1     iEth15          -          26        0      iEth03
    8        1     iEth16          -          26        1      iEth35
    8        2     iEth17          -          22        0      iEth32
    8        2     iEth18          -          22        1      iEth53
    8        2     iEth19          -          23        0      iEth32
    8        2     iEth20          -          23        1      iEth53
    8        2     iEth21          -          24        0      iEth32
    8        2     iEth22          -          24        1      iEth53
    8        2     iEth23          -          26        0      iEth32
    8        2     iEth24          -          26        1      iEth53
    8        3     iEth25          -          22        0      iEth31
    8        3     iEth26          -          22        1      iEth54
    8        3     iEth27          -          23        0      iEth31
    8        3     iEth28          -          23        1      iEth54
    8        3     iEth29          -          24        0      iEth31
    8        3     iEth30          -          24        1      iEth54
    8        3     iEth31          -          26        0      iEth31
    8        3     iEth32          -          26        1      iEth54
```

Similarly, iEth link mapping can be checked from a fabric module's perspective. An example of this is shown here, where internal links between the fabric module inserted in slot 22 and each of the 8 line cards installed in the Nexus 9508 chassis are displayed.

<#root>

Nexus9500#

show system internal fabric connectivity module 22

```
Internal Link-info Fabriccard slot:22
------------------------------------------------------------------------
 FM-Slot  FM-Unit   FM-iEthLink   LC-Slot   LC-Unit   LC-EthLink    MUX
------------------------------------------------------------------------
   22         0      iEth09          1         0      iEth01         -
   22         0      iEth06          1         1      iEth11         -
   22         0      iEth25          1         2      iEth21         -
   22         0      iEth26          1         3      iEth31         -
   22         0      iEth10          2         0      iEth01         -
   22         0      iEth05          2         1      iEth11         -
   22         0      iEth23          2         2      iEth21         -
   22         0      iEth24          2         3      iEth31         -
   22         0      iEth12          3         0      iEth01         -
   22         0      iEth11          3         1      iEth11         -
   22         0      iEth21          3         2      iEth21         -
   22         0      iEth22          3         3      iEth31         -
   22         0      iEth14          4         0      iEth01         -
   22         0      iEth13          4         1      iEth11         -
   22         0      iEth07          4         2      iEth21         -
   22         0      iEth08          4         3      iEth31         -
   22         0      iEth16          5         0      iEth01         -
   22         0      iEth15          5         1      iEth11         -
   22         0      iEth01          5         2      iEth21         -
   22         0      iEth04          5         3      iEth31         -
   22         0      iEth20          6         0      iEth01         -
   22         0      iEth17          6         1      iEth11         -
   22         0      iEth28          6         2      iEth21         -
   22         0      iEth27          6         3      iEth31         -
   22         0      iEth19          7         0      iEth01         -
```

```
  22         0      iEth02          7         1      iEth09         -
  22         0      iEth30          7         2      iEth17         -
  22         0      iEth29          7         3      iEth25         -
  22         0      iEth18          8         0      iEth01         -
  22         0      iEth03          8         1      iEth09         -
  22         0      iEth32          8         2      iEth17         -
  22         0      iEth31          8         3      iEth25         -
  22         1      iEth41          1         0      iEth02         -
  22         1      iEth38          1         1      iEth12         -
  22         1      iEth59          1         2      iEth22         -
  22         1      iEth60          1         3      iEth32         -
  22         1      iEth42          2         0      iEth02         -
  22         1      iEth37          2         1      iEth12         -
  22         1      iEth62          2         2      iEth22         -
  22         1      iEth61          2         3      iEth32         -
  22         1      iEth44          3         0      iEth02         -
  22         1      iEth43          3         1      iEth12         -
  22         1      iEth64          3         2      iEth22         -
  22         1      iEth63          3         3      iEth32         -
  22         1      iEth46          4         0      iEth02         -
  22         1      iEth45          4         1      iEth12         -
  22         1      iEth39          4         2      iEth22         -
  22         1      iEth40          4         3      iEth32         -
  22         1      iEth48          5         0      iEth02         -
  22         1      iEth47          5         1      iEth12         -
  22         1      iEth36          5         2      iEth22         -
  22         1      iEth33          5         3      iEth32         -
  22         1      iEth52          6         0      iEth02         -
  22         1      iEth49          6         1      iEth12         -
  22         1      iEth57          6         2      iEth22         -
  22         1      iEth58          6         3      iEth32         -
  22         1      iEth34          7         0      iEth02         -
  22         1      iEth51          7         1      iEth10         -
  22         1      iEth55          7         2      iEth18         -
  22         1      iEth56          7         3      iEth26         -
  22         1      iEth50          8         0      iEth02         -
  22         1      iEth35          8         1      iEth10         -
  22         1      iEth53          8         2      iEth18         -
  22         1      iEth54          8         3      iEth26         -
```

Use the **show system internal fabric link-state module {x}** command to check whether the internal port is up or not (under the ST columns), and what the corresponding ASIC slice and MAC identifier of a particular internal link (under the MAC column) are. An example of this is shown here.

<#root>

```
Nexus9500#
```

```
show system internal fabric link-state module 8
```

```
 cli : mod = 8
 module number = 8
================================================================================================================
 Module number = 8
================================================================================================================
 [LC]   [    INST:SLI:MAC:GLSRC]   [IETH]    [ST]   <========>  [FM]   [    INST:SLI:MAC:GLSRC]  [IETH]
================================================================================================================
 [ 8]   [  0 :   0 :   7 : 0x38]   [iEth01]   [UP]  <========>  [22]   [  0 :   3 :  21 :  0x18]   [iEt
```

```
[ 8]   [  0 :   1 :   9 :  0x0]   [iEth02]   [UP]  <========>  [22]   [  1 :   3 :  21 :  0x18]   [iEt
[ 8]   [  0 :   0 :   6 : 0x30]   [iEth03]   [UP]  <========>  [23]   [  0 :   3 :  21 :  0x18]   [iEt
[ 8]   [  0 :   1 :  16 : 0x38]   [iEth04]   [UP]  <========>  [23]   [  1 :   3 :  21 :  0x18]   [iEt
[ 8]   [  0 :   0 :   8 : 0x40]   [iEth05]   [UP]  <========>  [24]   [  0 :   3 :  21 :  0x18]   [iEt
[ 8]   [  0 :   1 :  15 : 0x30]   [iEth06]   [UP]  <========>  [24]   [  1 :   3 :  21 :  0x18]   [iEt
[ 8]   [  0 :   0 :   5 : 0x28]   [iEth07]   [UP]  <========>  [26]   [  0 :   3 :  21 :  0x18]   [iEt
[ 8]   [  0 :   1 :  17 : 0x40]   [iEth08]   [UP]  <========>  [26]   [  1 :   3 :  21 :  0x18]   [iEt
[ 8]   [  1 :   0 :   7 : 0x38]   [iEth09]   [UP]  <========>  [22]   [  0 :   0 :   4 :  0x20]   [iEt
[ 8]   [  1 :   1 :   9 :  0x0]   [iEth10]   [UP]  <========>  [22]   [  1 :   0 :   4 :  0x20]   [iEt
[ 8]   [  1 :   0 :   6 : 0x30]   [iEth11]   [UP]  <========>  [23]   [  0 :   0 :   4 :  0x20]   [iEt
[ 8]   [  1 :   1 :  16 : 0x38]   [iEth12]   [UP]  <========>  [23]   [  1 :   0 :   4 :  0x20]   [iEt
[ 8]   [  1 :   0 :   8 : 0x40]   [iEth13]   [UP]  <========>  [24]   [  0 :   0 :   4 :  0x20]   [iEt
[ 8]   [  1 :   1 :  15 : 0x30]   [iEth14]   [UP]  <========>  [24]   [  1 :   0 :   4 :  0x20]   [iEt
[ 8]   [  1 :   0 :   5 : 0x28]   [iEth15]   [UP]  <========>  [26]   [  0 :   0 :   4 :  0x20]   [iEt
[ 8]   [  1 :   1 :  17 : 0x40]   [iEth16]   [UP]  <========>  [26]   [  1 :   0 :   4 :  0x20]   [iEt
[ 8]   [  2 :   0 :   7 : 0x38]   [iEth17]   [UP]  <========>  [22]   [  0 :   5 :  35 :  0x28]   [iEt
[ 8]   [  2 :   1 :   9 :  0x0]   [iEth18]   [UP]  <========>  [22]   [  1 :   4 :  24 :   0x0]   [iEt
[ 8]   [  2 :   0 :   6 : 0x30]   [iEth19]   [UP]  <========>  [23]   [  0 :   5 :  35 :  0x28]   [iEt
[ 8]   [  2 :   1 :  16 : 0x38]   [iEth20]   [UP]  <========>  [23]   [  1 :   4 :  24 :   0x0]   [iEt
[ 8]   [  2 :   0 :   8 : 0x40]   [iEth21]   [UP]  <========>  [24]   [  0 :   5 :  35 :  0x28]   [iEt
[ 8]   [  2 :   1 :  15 : 0x30]   [iEth22]   [UP]  <========>  [24]   [  1 :   4 :  24 :   0x0]   [iEt
[ 8]   [  2 :   0 :   5 : 0x28]   [iEth23]   [UP]  <========>  [26]   [  0 :   5 :  35 :  0x28]   [iEt
[ 8]   [  2 :   1 :  17 : 0x40]   [iEth24]   [UP]  <========>  [26]   [  1 :   4 :  24 :   0x0]   [iEt
[ 8]   [  3 :   0 :   7 : 0x38]   [iEth25]   [UP]  <========>  [22]   [  0 :   5 :  34 :  0x20]   [iEt
[ 8]   [  3 :   1 :   9 :  0x0]   [iEth26]   [UP]  <========>  [22]   [  1 :   4 :  25 :   0x8]   [iEt
[ 8]   [  3 :   0 :   6 : 0x30]   [iEth27]   [UP]  <========>  [23]   [  0 :   5 :  34 :  0x20]   [iEt
[ 8]   [  3 :   1 :  16 : 0x38]   [iEth28]   [UP]  <========>  [23]   [  1 :   4 :  25 :   0x8]   [iEt
[ 8]   [  3 :   0 :   8 : 0x40]   [iEth29]   [UP]  <========>  [24]   [  0 :   5 :  34 :  0x20]   [iEt
[ 8]   [  3 :   1 :  15 : 0x30]   [iEth30]   [UP]  <========>  [24]   [  1 :   4 :  25 :   0x8]   [iEt
[ 8]   [  3 :   0 :   5 : 0x28]   [iEth31]   [UP]  <========>  [26]   [  0 :   5 :  34 :  0x20]   [iEt
[ 8]   [  3 :   1 :  17 : 0x40]   [iEth32]   [UP]  <========>  [26]   [  1 :   4 :  25 :   0x8]   [iEt
```

Step 2. Check the CRC counters on iEth Links and Track the Source of Corrupted Frames

On a modular Nexus 9500 switch, you can see CRC errors on one or more iEth links in these scenarios:

1. When the switch is operating in a a cut-through switching mode, a line card that receives a corrupted Ethernet frame with an incorrect CRC value in the FCS field does not drop the frame locally. Instead, the line card forwards the packet as normal. If the egress interface for the packet belongs to another ASIC or line card, the ingress line card forwards the packet towards a fabric module. The fabric modules also operate in a cut-through switching mode, so the fabric module forwards the packet to the egress line card. The egress line card forwards the packet towards the next-hop and increment the output errors counter on the egress interface.
2. If an internal link is failing due to faulty hardware, packets that traverse the internal link can be corrupted between a line card and the fabric module.

Use the **show system internal fabric connectivity stats module {x}** command to check the CRC counter of the corresponding internal links. An example of this is shown here, where the fabric module inserted in slot 22 receives packets with an invalid CRC on iEth56 connected to iEth26 of the line card inserted in slot 7 of the switch. This indicates that corrupted Ethernet frames are received by the fabric module from the line card inserted in slot 7 of the switch.

<#root>

```
Nexus9500#
```

show system internal fabric connectivity stats module 22

Internal Link-info Stats Fabriccard slot:22

FM-Slot	FM-Unit	FM-iEthLink	LC-Slot	LC-Unit	LC-EthLink	MUX	CRC
22	0	iEth09	1	0	iEth01	-	0
22	0	iEth06	1	1	iEth11	-	0
22	0	iEth25	1	2	iEth21	-	0
22	0	iEth26	1	3	iEth31	-	0
22	0	iEth10	2	0	iEth01	-	0
22	0	iEth05	2	1	iEth11	-	0
22	0	iEth23	2	2	iEth21	-	0
22	0	iEth24	2	3	iEth31	-	0
22	0	iEth12	3	0	iEth01	-	0
22	0	iEth11	3	1	iEth11	-	0
22	0	iEth21	3	2	iEth21	-	0
22	0	iEth22	3	3	iEth31	-	0
22	0	iEth14	4	0	iEth01	-	0
22	0	iEth13	4	1	iEth11	-	0
22	0	iEth07	4	2	iEth21	-	0
22	0	iEth08	4	3	iEth31	-	0
22	0	iEth16	5	0	iEth01	-	0
22	0	iEth15	5	1	iEth11	-	0
22	0	iEth01	5	2	iEth21	-	0
22	0	iEth04	5	3	iEth31	-	0
22	0	iEth20	6	0	iEth01	-	0
22	0	iEth17	6	1	iEth11	-	0
22	0	iEth28	6	2	iEth21	-	0
22	0	iEth27	6	3	iEth31	-	0
22	0	iEth19	7	0	iEth01	-	0
22	0	iEth02	7	1	iEth09	-	0
22	0	iEth30	7	2	iEth17	-	0
22	0	iEth29	7	3	iEth25	-	0
22	0	iEth18	8	0	iEth01	-	0
22	0	iEth03	8	1	iEth09	-	0
22	0	iEth32	8	2	iEth17	-	0
22	0	iEth31	8	3	iEth25	-	0
22	1	iEth41	1	0	iEth02	-	0
22	1	iEth38	1	1	iEth12	-	0
22	1	iEth59	1	2	iEth22	-	0
22	1	iEth60	1	3	iEth32	-	0
22	1	iEth42	2	0	iEth02	-	0
22	1	iEth37	2	1	iEth12	-	0
22	1	iEth62	2	2	iEth22	-	0
22	1	iEth61	2	3	iEth32	-	0
22	1	iEth44	3	0	iEth02	-	0
22	1	iEth43	3	1	iEth12	-	0
22	1	iEth64	3	2	iEth22	-	0
22	1	iEth63	3	3	iEth32	-	0
22	1	iEth46	4	0	iEth02	-	0
22	1	iEth45	4	1	iEth12	-	0
22	1	iEth39	4	2	iEth22	-	0
22	1	iEth40	4	3	iEth32	-	0
22	1	iEth48	5	0	iEth02	-	0
22	1	iEth47	5	1	iEth12	-	0
22	1	iEth36	5	2	iEth22	-	0
22	1	iEth33	5	3	iEth32	-	0
22	1	iEth52	6	0	iEth02	-	0
22	1	iEth49	6	1	iEth12	-	0
22	1	iEth57	6	2	iEth22	-	0
22	1	iEth58	6	3	iEth32	-	0

```
  22         1      iEth34          7         0      iEth02         -     0
  22         1      iEth51          7         1      iEth10         -     0
  22         1      iEth55          7         2      iEth18         -     0
```

22

1

iEth56 7

3

iEth26

-

1665601166

```
  22         1      iEth50          8         0      iEth02         -     0
  22         1      iEth35          8         1      iEth10         -     0
  22         1      iEth53          8         2      iEth18         -     0
  22         1      iEth54          8         3      iEth26         -     0
```

Use the **slot {x} show hardware internal tah counters asic {y}** command on a line card or fabric module to determine if CRC errors are invalid or stomped CRCs. The two register counters that differentiate invalid CRC errors from stomped CRC errors are:

1. 02-RX Frm with FCS Err - Indicates a frame with an invalid, but non-stomped CRC was received.
2. 16-RX Frm CRC Err(Stomp) - Indicates a frame with a stomped CRC was received.

An example of this is shown here, where corrupted frames are received on the fabric module inserted in slot 22 of the chassis through internal link iEth54 connect to the line card inserted in slot 8of the chassis are received with stomped CRC:

<#root>

Nexus9500#

slot 22 show hardware internal tah counters asic 1

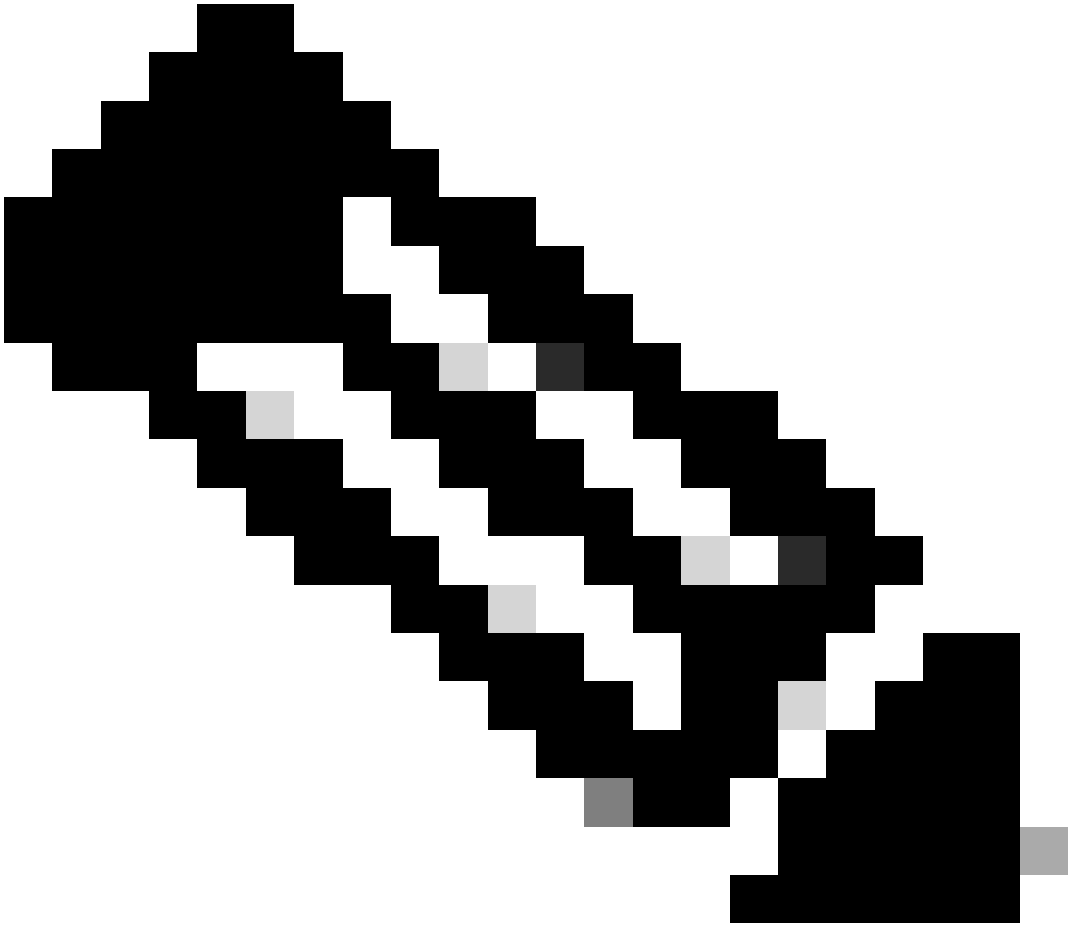
```
REG_NAME                  M24,0-100Gx4
-------------------------------------------------------------------------------------------------------------
02-RX Frm with FCS Err    ....
03-RX Frm with any Err    ....
16-RX
```

Frm CRC Err(Stomp)

....

14491277d7

Alternatively, use the **show hardware internal errors module {x}** command to see ASIC error counters for a specific module. An example of this is shown here.



Note: In this output, the Interface Inbound Errors (CRC,len,Algn Err) counter increments for both invalid CRCs and stomped CRCs, while the Interface Inbound CRC Error Stomped counter increments for only stomped CRCs.

<#root>

Nexus9500#

show hardware internal errors module 22

```
|-----------------------------------------------------------------------|
| Device:Lacrosse                 Role:MAC                     Mod:22    |
| Last cleared @ Tue Jul  6 04:10:45 2021
| Device Statistics Category :: ERROR
|-----------------------------------------------------------------------|
Instance:0
ID    Name                                          Value             Ports
--    ----                                          -----             -----

Instance:
```

1

```
ID    Name                                          Value            Ports
--    ----                                          -----            -----
196635  Interface Inbound Errors (CRC,len,Algn Err)

0000053053264536


27:0


1048603 Interface Inbound CRC Error Stomped

0000053053264535


27:0
```

After identifying the ingress line card from which the corrupted frames are being received, use the **slot {x} show hardware internal tah counters asic {y}** or **show hardware internal errors module {x}** commands in a similar way to identify the ingress interface that the errors are received on, as well as if errors are received as invalid CRCs or stomped CRCs.

A rare scenario is possible where a fabric module or egress line card shows CRC errors on an iEth link, but the connected line card has no signs of ingress CRCs. The root cause of this issue is typically hardware failure of the fabric module. Cisco recommends opening a support case with Cisco TAC to troubleshoot this issue further and replace the fabric module if necessary.

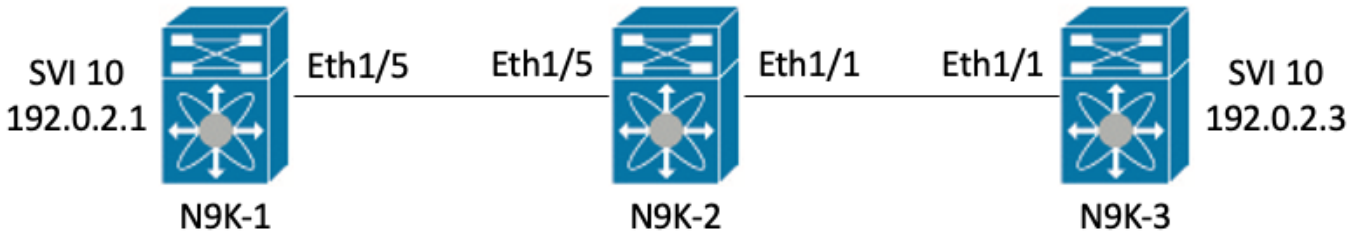
Examples

This section of the document walks through the previous procedure with some examples.

Scenario 1. Physical Interface Receiving Stomped CRCs

This example demonstrates how to identify that CRC errors on a physical interface are stomped CRCs.

Consider this topology:



In this example, purposefully-stomped CRC errors are generated on switch N9K-1 through jumbo-sized 8000 byte ICMP packets sourced from interface SVI 10 (which owns IP address 192.0.2.1) destined to N9K-3's interface SVI 10 (which owns IP address 192.0.2.3), which has an MTU of 1500 bytes. N9K-1, N9K-2, and N9K-3 are all Nexus 93180YC-EX model switches.

<#root>

```
N9K-1#
```

```
ping 192.0.2.3 count 5 packet-size 8000
```

```
PING 192.0.2.3 (192.0.2.3): 8000 data bytes
Request 0 timed out
Request 1 timed out
Request 2 timed out
Request 3 timed out
Request 4 timed out
Request 5 timed out

--- 192.0.2.3 ping statistics ---
5 packets transmitted, 0 packets received, 100.00% packet loss
```

In this example, incrementing CRC errors are observed on physical interface Ethernet1/1 of switch N9K-3.

<#root>

```
N9K-3#
```

```
show interface Ethernet1/1
```

```
<snip>
Ethernet1/1 is up
admin state is up, Dedicated Interface
  Hardware: 100/1000/10000/25000 Ethernet, address: 00d7.8f86.2bbe (bia 00d7.8f86.2bbe)
  MTU 1500 bytes, BW 10000000 Kbit, DLY 10 usec
  reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, medium is broadcast
  Port mode is trunk
  full-duplex, 10 Gb/s, media type is 10G
  Beacon is turned off
  Auto-Negotiation is turned on  FEC mode is Auto
  Input flow-control is off, output flow-control is off
  Auto-mdix is turned off
  Rate mode is dedicated
  Switchport monitor is off
  EtherType is 0x8100
  EEE (efficient-ethernet) : n/a
    admin fec state is auto, oper fec state is off
  Last link flapped 06:13:44
  Last clearing of "show interface" counters 02:55:00
  0 interface resets
  RX
    9 unicast packets  10675 multicast packets  0 broadcast packets
    10691 input packets  816924 bytes
    7 jumbo packets  0 storm suppression bytes
    0 runts  7 giants
```

```
7 CRC
```

```
  0 no buffer
    7 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  0 input discard
    0 Rx pause
```

Step 1. Confirm Incrementing CRCs

Confirm that CRCs are incrementing on physical interface Ethernet1/1 by generating jumbo-sized 8000 byte ICMP packets sourced from N9K-1's interface SVI 10 (which owns IP address 192.0.2.1) destined to N9K-3's interface SVI 10 (which owns IP address 192.0.2.3).

```
<#root>

N9K-1#

ping 192.0.2.3 count 5 packet-size 8000

PING 192.0.2.3 (192.0.2.3): 8000 data bytes
Request 0 timed out
Request 1 timed out
Request 2 timed out
Request 3 timed out
Request 4 timed out
Request 5 timed out

--- 192.0.2.3 ping statistics ---
5 packets transmitted, 0 packets received, 100.00% packet loss

N9K-3#

show interface Ethernet1/1


Ethernet1/1 is up
admin state is up, Dedicated Interface
  Hardware: 100/1000/10000/25000 Ethernet, address: 00d7.8f86.2bbe (bia 00d7.8f86.2bbe)
  MTU 1500 bytes, BW 10000000 Kbit, DLY 10 usec
  reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, medium is broadcast
  Port mode is trunk
  full-duplex, 10 Gb/s, media type is 10G
  Beacon is turned off
  Auto-Negotiation is turned on  FEC mode is Auto
  Input flow-control is off, output flow-control is off
  Auto-mdix is turned off
  Rate mode is dedicated
  Switchport monitor is off
  EtherType is 0x8100
  EEE (efficient-ethernet) : n/a
    admin fec state is auto, oper fec state is off
  Last link flapped 06:52:57
  Last clearing of "show interface" counters 03:34:13
  0 interface resets
  RX
    11 unicast packets  13066 multicast packets  0 broadcast packets
    13089 input packets  1005576 bytes
    12 jumbo packets  0 storm suppression bytes
    0 runts  12 giants

12 CRC

  0 no buffer
    12 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  0 input discard
    0 Rx pause
```

Step 2. Map Physical Interface to ASIC, MAC Block, and MAC Block Sub-Port

Use the **show interface hardware-mappings** command on N9K-3 to map physical interface Ethernet1/1 to ASIC number 0, MAC block 4, and MAC block sub-port 0.

```
<#root>

N9K-3#

show interface hardware-mappings


<snip>
-------------------------------------------------------------------------------------------------------------
Name       Ifindex  Smod

Unit

 HPort FPort NPort VPort Slice SPort SrcId

MacId MacSP

 VIF  Block BlkSrcID
-------------------------------------------------------------------------------------------------------------

Eth1/1

     1a000000 1

0

    16    255   0     -1    0     16    32

4     0

     1    0     32
Eth1/2     1a000200 1    0    17    255   4     -1    0     17    34    4     2     5     0     34
Eth1/3     1a000400 1    0    18    255   8     -1    0     18    36    4     4     9     0     36
Eth1/4     1a000600 1    0    19    255   12    -1    0     19    38    4     6     13    0     38
Eth1/5     1a000800 1    0    12    255   16    -1    0     12    24    3     0     17    0     24
```

Step 3. Check Cloud Scale ASIC Registers for CRC-Related Counters

Based on the information from Step 2, you know these facts:

1. Physical interface Ethernet1/1 is mapped to ASIC number 0.
2. Physical interface Ethernet1/1 is mapped to MAC block sub-port 0 of MAC block 4
3. Because N9K-3 is a top-of-rack Nexus 93180YC-EX model switch, you know that the only possible line card slot number is 1
4. From the output of show interface gathered in Step 1, you know the speed of physical interface Ethernet1/1 is 10G.

Using this information, you can use the **slot 1 show hardware internal tah counters asic 0** command to view the ASIC register counters for all physical interfaces. Specifically, you are looking for ASIC register counters associated with M4,0-10G.

```
<#root>
```

```
N9K-3#
```

slot 1 show hardware internal tah counters asic 0

```
<snip>
*************** PER MAC/CH SRAM COUNTERS ****************
REG_NAME
```

M4,0-10G

```
     M4,2-10G     M4,4-10G     M4,6-10G     M5,0-40Gx4     M6,0-40Gx4     M7,0-40Gx4     M8,0-10G
-------------------------------------------------------------------------------------------------
02-RX Frm with FCS Err   ....         ....         ....         ....         ....           ....
```

16-RX Frm CRC Err(Stomp) c

```
            ....         ....         ....         ....           ....           ....           ....
```

You can see a non-zero hexadecimal value of 0xc for register 16, which indicates frames with a stomped CRC were received on this physical interface. You can use the **dec 0xc** command to translate this to a decimal value of 12, which matches the number of CRC errors on physical interface Ethernet1/1.

<#root>

N9K-3#

dec 0xc

12

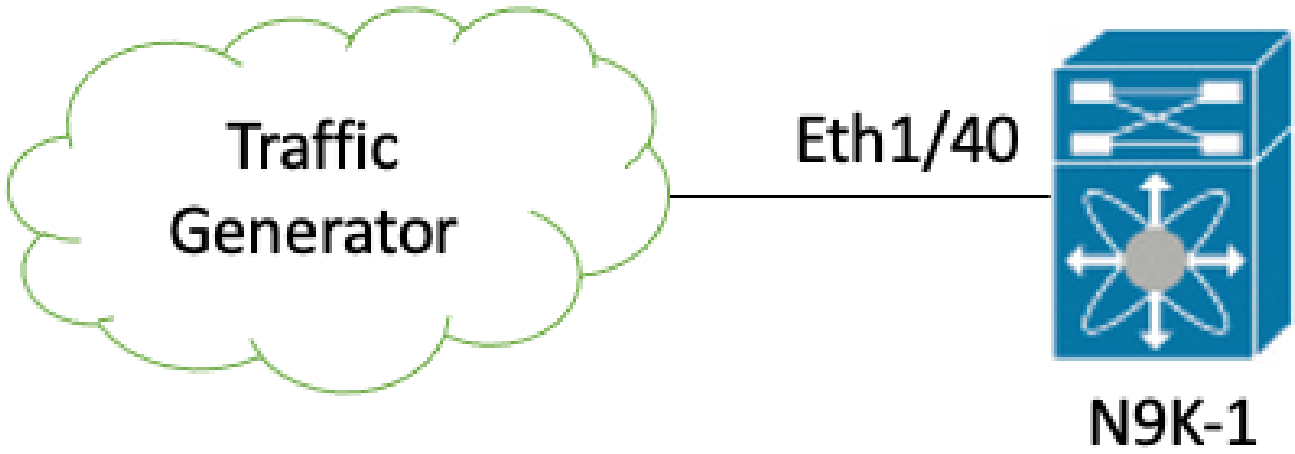
Conclusion

You have confirmed that N9K-3 is receiving frames with a stomped CRC on physical interface Ethernet1/1. This means that the device on the remote side of the Ethernet1/1 link (in this case, N9K-2) is stomping the CRC of these frames; the root cause of the malformed frames is not the link directly connected to Ethernet1/1, but is further downstream. Additional troubleshooting can be performed on the downstream network device to determine the source of these malformed frames.

Scenario 2. Physical Interface Received Malformed Frames with Invalid CRC

This example demonstrates how to identify that CRC errors on a physical interface are incrementing due to malformed frames caused by a physical layer issue on a directly-connected link.

Consider this topology:



In this example, a traffic generator connected to physical interface Ethernet1/40 of switch N9K-1 is purposefully generating frames with an incorrect CRC. This simulates a physical layer issue on the link connected to Ethernet1/40, such as a faulty transceiver or damaged cable. N9K-1 receives these frames, recognizes that the CRC is invalid, and increments the CRC error counter on the Ethernet1/40 physical interface. N9K-1 is a Nexus 93180YC-EX model switch.

<#root>

N9K-1#

show interface Ethernet1/40

```
Ethernet1/40 is up
admin state is up, Dedicated Interface
  Hardware: 100/1000/10000/25000 Ethernet, address: 00d7.8f86.2bbe (bia 00d7.8f86.2c02)
  MTU 1500 bytes, BW 10000000 Kbit, DLY 10 usec
  reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, medium is broadcast
  Port mode is trunk
  full-duplex, 10 Gb/s, media type is 10G
  Beacon is turned off
  Auto-Negotiation is turned on  FEC mode is Auto
  Input flow-control is off, output flow-control is off
  Auto-mdix is turned off
  Rate mode is dedicated
  Switchport monitor is off
  EtherType is 0x8100
  EEE (efficient-ethernet) : n/a
    admin fec state is auto, oper fec state is off
  Last link flapped 06:13:44
  Last clearing of "show interface" counters 02:55:00
  0 interface resets
  RX
    1710 unicast packets  9873 multicast packets  0 broadcast packets
    11583 input packets  886321 bytes
    0 jumbo packets  0 storm suppression bytes
    0 runts  0 giants
```

1683 CRC

```
  0 no buffer
```

```
1683 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  0 input discard
    0 Rx pause
```

Step 1. Confirm Incrementing CRCs

Confirm that CRCs are incrementing on physical interface Ethernet1/40 of N9K-1 through the **show interface** or **show interface counters non-zero** commands.

<#root>

```
N9K-1#

show interface Ethernet1/40

<snip>
Ethernet1/40 is up
admin state is up, Dedicated Interface
  Hardware: 100/1000/10000/25000 Ethernet, address: 00d7.8f86.2bbe (bia 00d7.8f86.2c02)
  MTU 1500 bytes, BW 10000000 Kbit, DLY 10 usec
  reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, medium is broadcast
  Port mode is trunk
  full-duplex, 10 Gb/s, media type is 10G
  Beacon is turned off
  Auto-Negotiation is turned on  FEC mode is Auto
  Input flow-control is off, output flow-control is off
  Auto-mdix is turned off
  Rate mode is dedicated
  Switchport monitor is off
  EtherType is 0x8100
  EEE (efficient-ethernet) : n/a
    admin fec state is auto, oper fec state is off
  Last link flapped 06:13:44
  Last clearing of "show interface" counters 02:55:00
  0 interface resets
  RX
    14055 unicast packets  9873 multicast packets  0 broadcast packets
    23928 input packets  1676401 bytes
    0 jumbo packets  0 storm suppression bytes
    0 runts  0 giants

14028 CRC

  0 no buffer


14028 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  0 input discard
    0 Rx pause

N9K-1#

show interface counters errors non-zero
```

```
<snip>
--------------------------------------------------------------------------------
Port
          Align-Err
    FCS-Err
   Xmit-Err    Rcv-Err  UnderSize OutDiscards
--------------------------------------------------------------------------------
Eth1/40
26373
      26373
          0      26373          0           0
```

Step 2. Map Physical Interface to ASIC, MAC Block, and MAC Block Sub-Port

Use the **show interface hardware-mappings** command on N9K-1 to map physical interface Ethernet1/40 to ASIC number 0, MAC block 10, MAC block sub-port 6.

```
<#root>
N9K-1#
show interface hardware-mappings
<snip>
-----------------------------------------------------------------------------------------------
Name       Ifindex  Smod
Unit
 HPort FPort NPort VPort Slice SPort SrcId
MacId

MacSP
 VIF  Block BlkSrcID
-----------------------------------------------------------------------------------------------
Eth1/38    1a004a00 1    0    45    255   148   -1    1     5     10    10    2     149  0     10
Eth1/39    1a004c00 1    0    46    255   152   -1    1     6     12    10    4     153  0     12
Eth1/40
    1a004e00 1
0
    47    255   156   -1    1     7     14
10

6
```

```
     157  0     14
Eth1/41    1a005000 1    0    76    255   160   -1    1     36    64    17    0     161  0     64
Eth1/42    1a005200 1    0    77    255   164   -1    1     37    66    17    2     165  0     66
```

Step 3. Check Cloud Scale ASIC Registers for CRC-Related Counters

Based on the information from Step 2, you know these facts:

1. Physical interface Ethernet1/40 is mapped to ASIC number 0.
2. Physical interface Ethernet1/40 is mapped to MAC block sub-port 6 of MAC block 10.
3. Because N9K-1 is a top-of-rack Nexus 93180YC-EX model switch, you know that the only possible line card slot number is 1.
4. From the output of **show interface** gathered in Step 1, you know the speed of physical interface Ethernet1/40 is 10G.

Using this information, you can use the slot 1 show hardware internal tah counters asic 0 command to view the ASIC register counters for all physical interfaces. Specifically, you are looking for ASIC register counters associated with M10,6-10G.

```
<#root>

N9K-1#

slot 1 show hardware internal tah counters asic 0


*************** PER MAC/CH SRAM COUNTERS ****************
REG_NAME                 M8,2-10G     M8,4-10G     M8,6-10G     M9,0-40Gx4     M10,0-10G      M10,2-10G

M10,6-10G

-------------------------------------------------------------------------------------------------------------

02-RX Frm with FCS Err

   ....         ....         ....         ....           ....           ....           ....

973e


16-RX Frm CRC Err(Stomp) ....         ....         ....         ....           ....           ....
```

You can see a non-zero hexadecimal value of 0x973e for register 2, which indicates frames with an invalid, but non-stomped CRC were received on this physical interface.

You can use the dec 0x973e command to translate this to a decimal value of 38,718, which matches (or is less than, since the CRCs are constantly incrementing) the number of CRC errors on physical interface Ethernet1/40.

```
<#root>

N9K-1#

dec 0x973e
```

```
38718
```

Conclusion

You have confirmed that N9K-1 is receiving frames with an invalid, but non-stomped CRC on physical interface Ethernet1/40. This means that the link directly connected to Ethernet1/40 (or the device on the remote end of the link) is the most likely source of the malformed frames. Further troubleshooting can be performed on the physical layer of this link to isolate the root cause of the malformed frames (such as checking for damaged cabling, replacing the current transceivers with known good transceivers, and so on).

Scenario 3. Nexus 9500 iEth CRC Errors Syslog

This example demonstrates how to identify the source of CRC errors on an iEth internal link when a syslog reporting errors on an internal interface is generated by a Nexus 9500 series switch. An example of this syslog is shown here.

```
<#root>

Nexus9500#

show logging logfile


<snip>
2021 Jul  9 05:51:19 Nexus9500 %DEVICE_TEST-SLOT22-3-INTERNAL_PORT_MONITOR_CRC_ERRORS_DETECTED: Module
```

This syslog indicates that errors were detected on the iEth56 internal link of the fabric module inserted in slot 22 of the switch.

Step 1. Map iEth Link on Fabric Module to Connected Line Card

Use the **show system internal fabric connectivity stats module {x}** command to identify which line card the affected iEth internal link connects to. In this example, iEth56 of the fabric module inserted in slot 22 of the switch has errors. An example of this is shown here, where iEth56 of the fabric module inserted in slot 22 is connected to iEth26 of the line card inserted in slot 7 of the switch.

```
<#root>

Nexus9500#

show system internal fabric connectivity stats module 22 | include Eth56|FM-Slot


FM-Slot  FM-Unit   FM-iEthLink   LC-Slot   LC-Unit   LC-EthLink    MUX    CRC
   22          1      iEth56

 7

        3

iEth26
```

```
         -
603816174
```

Use the **show system internal fabric link-state module {x}** command to locate the ASIC instance and MAC identifier associated with the fabric module's iEth56 internal link. An example of this is shown here, where the ASIC instance is 1 and the MAC identifier is 27.

```
<#root>
Nexus9500#
show system internal fabric link-state module 22 | include MAC|iEth56

 [FM]   [
INST
:SLI:
MAC
:GLSRC]   [IETH]    [ST]   <========>  [LC]   [    INST:SLI:MAC:GLSRC]  [IETH]   [ST]
 [22]   [
1
 :   4 :
27
 : 0x18]   [iEth56]   [UP]  <========>  [ 7]   [  3 :   1 :   9 :   0x0]   [iEth26]   [UP]
```

Step 2. Check if CRCs Received on iEth Link are Invalid or Stomped

The previous step shows that our ASIC instance identifier is 1 and our MAC identifier is 27 for iEth56 connected to the fabric module inserted in slot 22. Use the **slot {x} show hardware internal tah counters asic {y}** command to identify whether the CRCs reported by the syslog are invalid CRCs or stomped CRCs. An example of this is shown here, where the M27,0-100Gx4 column is associated with our MAC identifier of 27 and indicates the CRCs are stomped.

```
<#root>
Nexus9500#
slot 22 show hardware internal tah counters asic 1

REG_NAME                  M
27
,0-100Gx4
----------------------------------------
02-RX Frm with FCS Err    ....
16-RX Frm CRC Err(Stomp)
be9cb9bd6
```

Alternatively, use the **show hardware internal errors module {x}** command to glean this same information. An example of this is shown here.

```
<#root>

Nexus9500#

show hardware internal errors module 22 | include CRC|Stomp|Inst


Instance:

1


196635 Interface Inbound Errors (CRC,len,Algn Err)    0000051587084851

27:0


1048603 Interface Inbound CRC Error Stomped           0000051587084850

27:0
```

Recall that in this output, the Interface Inbound Errors (CRC,len,Algn Err) counter increments for both invalid CRCs and stomped CRCs, while the Interface Inbound CRC Error Stomped counter increments for only stomped CRCs.

Step 3. Track Source of Frames with Invalid CRCs on Ingress Line Card

You now know that the CRCs entering the fabric module inserted in slot 22 of the switch are entering the switch from the line card inserted in slot 7. With this information, you can use the **show interface counters errors module {x} non-zero** command to identify non-zero CRC counters on interfaces belonging to the relevant line card. An example of this is shown here.

```
<#root>

Nexus9500#

show interface counters errors module 7 non-zero


<snip>
--------------------------------------------------------------------------------
Port          Align-Err    FCS-Err   Xmit-Err    Rcv-Err  UnderSize OutDiscards
--------------------------------------------------------------------------------

Eth7/32

               0          0          0

1195309745

          0           0
```

You can repeat Step #2 of this scenario on the relevant line card to verify whether the line card is receiving invalid CRCs or stomped CRCs.

```
<#root>

Nexus9500#

show hardware internal errors module 7 | include ignore-case CRC|Stomp|Inst


Instance:3


196619 Interface Inbound Errors (CRC,len,Algn Err)

0000051801011139


11:0


1048587 Interface Inbound CRC Error Stomped

0000051801011140


11:0
```

Use the **show interface hardware-mappings** command to identify the front-panel port that the MacId:MacSP value of 11:0 in the previous output is mapped to. An example of this is shown here, where 11:0 maps to front-panel port Eth7/32.

```
<#root>

Nexus9500#

show interface hardware-mappings | include Name|Eth7


<snip>
Name       Ifindex  Smod

Unit

 HPort FPort NPort VPort Slice SPort SrcId

MacId MacSP

 VIF  Block BlkSrcID
Eth7/1     1a300000 25   0    16    255   0     -1    0     16    32    4     0     1     0     32
Eth7/2     1a300200 25   0    12    255   4     -1    0     12    24    3     0     5     0     24
Eth7/3     1a300400 25   0    8     255   8     -1    0     8     16    2     0     9     0     16
Eth7/4     1a300600 25   0    4     255   12    -1    0     4     8     1     0     13    0     8
Eth7/5     1a300800 25   0    60    255   16    -1    1     20    40    14    0     17    0     40
Eth7/6     1a300a00 25   0    56    255   20    -1    1     16    32    13    0     21    0     32
Eth7/7     1a300c00 25   0    52    255   24    -1    1     12    24    12    0     25    0     24
Eth7/8     1a300e00 25   0    48    255   28    -1    1     8     16    11    0     29    0     16
Eth7/9     1a301000 26   1    12    255   32    -1    0     12    24    3     0     33    0     24
Eth7/10    1a301200 26   1    8     255   36    -1    0     8     16    2     0     37    0     16
```

```
Eth7/11     1a301400 26   1     4      255    40     -1     0      4      8      1      0      41     0      8
Eth7/12     1a301600 26   1     0      255    44     -1     0      0      0      0      0      45     0      0
Eth7/13     1a301800 26   1     60     255    48     -1     1      20     40     14     0      49     0      40
Eth7/14     1a301a00 26   1     56     255    52     -1     1      16     32     13     0      53     0      32
Eth7/15     1a301c00 26   1     52     255    56     -1     1      12     24     12     0      57     0      24
Eth7/16     1a301e00 26   1     48     255    60     -1     1      8      16     11     0      61     0      16
Eth7/17     1a302000 27   2     16     255    64     -1     0      16     32     4      0      65     0      32
Eth7/18     1a302200 27   2     12     255    68     -1     0      12     24     3      0      69     0      24
Eth7/19     1a302400 27   2     8      255    72     -1     0      8      16     2      0      73     0      16
Eth7/20     1a302600 27   2     4      255    76     -1     0      4      8      1      0      77     0      8
Eth7/21     1a302800 27   2     60     255    80     -1     1      20     40     14     0      81     0      40
Eth7/22     1a302a00 27   2     56     255    84     -1     1      16     32     13     0      85     0      32
Eth7/23     1a302c00 27   2     52     255    88     -1     1      12     24     12     0      89     0      24
Eth7/24     1a302e00 27   2     48     255    92     -1     1      8      16     11     0      93     0      16
Eth7/25     1a303000 28   3     12     255    96     -1     0      12     24     3      0      97     0      24
Eth7/26     1a303200 28   3     8      255    100    -1     0      8      16     2      0      101    0      16
Eth7/27     1a303400 28   3     4      255    104    -1     0      4      8      1      0      105    0      8
Eth7/28     1a303600 28   3     0      255    108    -1     0      0      0      0      0      109    0      0
Eth7/29     1a303800 28   3     60     255    112    -1     1      20     40     14     0      113    0      40
Eth7/30     1a303a00 28   3     56     255    116    -1     1      16     32     13     0      117    0      32
Eth7/31     1a303c00 28   3     52     255    120    -1     1      12     24     12     0      121    0      24

Eth7/32

    1a303e00 28

3

    48    255   124   -1    1     8     16

11    0

     125  0     16
```

Conclusion

You have confirmed that the Nexus 9500 is receiving frames with a stomped CRC on physical interface Ethernet7/32. This means that the device on the remote side of the Ethernet7/32 link is stomping the CRC of these frames; the root cause of the malformed frames is not the link directly connected to Ethernet7/32, but is further downstream. Additional troubleshooting can be performed on the downstream network device to determine the source of these malformed frames.

Scenario 4. Track Source of Invalid CRC Frames with Egress Interface

This example demonstrates how to track the source of frames with invalid CRCs on a Nexus 9500 switch when an upstream switch reports that the Nexus 9500 is generating frames with stomped CRCs. In this scenario, the upstream switch is connected via front-panel port Ethernet8/9.

Step 1. Identify Fabric Module Sending Invalid CRC Frames to Egress Line Card

You know that the egress interface sending frames with stomped CRCs towards the upstream switch is Ethernet8/9. First, you need to determine the fabric module that is sending frames with stomped CRCs to the line card inserted in slot 8 of the chassis. You start this process with the **show hardware internal errors module {x}** command. An example of this is shown here.

<#root>

```
Nexus9500#
show hardware internal errors module 8 | i CRC|Inst


<snip>
Instance:1
196617 Interface Inbound Errors (CRC,len,Algn Err)    0000091499464650
9:0


1048585 Interface Inbound CRC Error Stomped           0000091499464651
9:0
```

MacID:MacSP 9:0 in the prevous output can be mapped to the source fabric module with the **show system internal fabric link-state module 8** command. An example of this is shown here.

```
<#root>
Nexus9500#
show system internal fabric link-state module 8


 cli : mod = 8
 module number = 8

================================================================================================================
 Module number = 8
================================================================================================================
 [LC]   [
INST
:SLI:
MAC
:GLSRC]   [IETH]    [ST]   <========>  [FM]   [    INST:SLI:MAC:GLSRC]  [IETH]   [ST]
================================================================================================================
 ...
[ 8]   [
1
 :   1 :
9
 :  0x0]   [iEth10]   [UP]  <========>  [
22
]   [  1 :   0 :   4 :  0x20]   [iEth35]   [UP]
```

You see that MAC identifier 9 on the line card inserted in slot 8 is mapped to the fabric module inserted in slot 22 of the chassis. You expect to see CRC errors on internal link iEth10. You can validate this with the **show system internal fabric connectivity stats module 8** command. An example of this is shown here.

```
<#root>

Nexus9500#

show system internal fabric connectivity stats module 8


Internal Link-info Stats Linecard slot:8
------------------------------------------------------------------------------------
 LC-Slot  LC-Unit   LC-iEthLink   MUX   FM-Slot   FM-Unit   FM-iEthLink    CRC
------------------------------------------------------------------------------------
    8        0     iEth01        -        22        0      iEth18        0
    8        0     iEth02        -        22        1      iEth50        0
    8        0     iEth03        -        23        0      iEth18        0
    8        0     iEth04        -        23        1      iEth50        0
    8        0     iEth05        -        24        0      iEth18        0
    8        0     iEth06        -        24        1      iEth50        0
    8        0     iEth07        -        26        0      iEth18        0
    8        0     iEth08        -        26        1      iEth50        0
    8        1     iEth09        -        22        0      iEth03        0
    8        1     iEth10        -        22        1      iEth35

1784603561
```

Step 2. Map iEth Link on Fabric Module to the Connected Linecard and Check for Stomped CRCs

Next, you use the same process as in Scenario 3 by checking the iEth internal links that receive CRCs, whether those CRCs are stomped or not according to the fabric module's ASIC, and what line card is connected to the fabric module's iEth internal link. An example of this is shown here using the **show system internal fabric connectivity stats module {x}** command, the **show hardware internal errors module {x}** command, and the **show system internal fabric link-state module {x}** command, respectively.

```
<#root>

Nexus9500#

show system internal fabric connectivity stats module 22


Internal Link-info Stats Fabriccard slot:22
------------------------------------------------------------------------------------
 FM-Slot  FM-Unit   FM-iEthLink   LC-Slot   LC-Unit   LC-EthLink    MUX    CRC
    22        1

iEth56

          7         3      iEth26         -

1171851894



Nexus9500#

show hardware internal errors module 22 | i CRC|Stomp|Inst


Instance:1
196635  Interface Inbound Errors (CRC,len,Algn Err)   0000054593935847
```

```
27:0

1048603 Interface Inbound CRC Error Stomped           0000054593935846

27:0


Nexus9500#

show system internal fabric link-state module 22 | i MAC|iEth56


 [FM]   [    INST:SLI:MAC:GLSRC]   [IETH]    [ST]   <========>  [LC]   [    INST:SLI:MAC:GLSRC]  [IETH]
 [22]   [  1 :   4 :  27 : 0x18]   [iEth56]   [UP]  <========>  [ 7]   [  3 :   1 :   9 :   0x0]   [iEt
```

Step 3. Track Source of Frames with Invalid CRCs on Ingress Module

After determining the ingress line card (in this scenario, the line card inserted in slot 7 connected by iEth26 to iEth56 of the fabric module inserted in slot 22), you identify which ingress port the corrupted frames enter the switch. This is done with the **show interface counters errors module {x} non-zero** command. The output of the show hardware internal errors module {x} command and the show interface hardware-mappings command can validate whether the received frames are invalid or stomped CRCs.

An example of this is shown here, where corrupted frames are entering the switch through front-panel interface Ethernet7/32.

```
<#root>

Nexus9500#

show interface counters errors module 7 non-zero


<snip>
--------------------------------------------------------------------------------
Port          Align-Err    FCS-Err   Xmit-Err    Rcv-Err  UnderSize OutDiscards
--------------------------------------------------------------------------------
Eth7/32               0          0          0 4128770335          0           0

--------------------------------------------------------------------------------
Port         Stomped-CRC
--------------------------------------------------------------------------------
Eth7/32      4129998971

Nexus9500#

show hardware internal errors module 7 | i i CRC|Stomp|Inst


<snip>

Instance:3


196619  Interface Inbound Errors (CRC,len,Algn Err)   0000054901402307

11:0
```

```
1048587 Interface Inbound CRC Error Stomped           0000054901402308
11:0


Nexus9500#
show interface hardware-mappings | i Name|Eth7


<snip>
Name       Ifindex  Smod
Unit
 HPort FPort NPort VPort Slice SPort SrcId
MacId


MacSP
 VIF  Block BlkSrcID
...
Eth7/32
    1a303e00 28
3
    48    255   124   -1    1     8     16
11


0
     125  0     16
```

Conclusion

You have confirmed that the Nexus 9500 is receiving frames with a stomped CRC on physical interface Ethernet7/32. This means that the device on the remote side of the Ethernet7/32 link is stomping the CRC of these frames; the root cause of the malformed frames is not the link directly connected to Ethernet7/32, but is further downstream.

Additional troubleshooting can be performed on the downstream network device to determine the source of these malformed frames.

Related Information

- Cut-Through and Store-and-Forward Ethernet Switching for Low-Latency Environments