Nexus 9000 Cloud Scale ASIC (Tahoe) NX-OS ELAM

Contents

Introduction

This document describes the steps used in order to perform an ELAM (Embedded Logic Analyzer Module) on a series of Cisco Nexus 9000 CloudScale ASIC modules, covers the most relevant outputs, and describes how to interpret the results.

> **Tip**: Refer to the ELAM Overview document for an overview on ELAM.

Applicable Hardware

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

N9K-C93180YC-EX	N9K-C92304QC
N9K-X9736C-EX	N9K-C92300YC
N9K-C93108TC-EX	N9K-X9788TC-FX
N9K-X9732C-EX	N9K-X97284YC-FX
N9K-X97160YC-EX	N9K-C93180YC-FX
N9K-C93180LC-EX	N9K-C93108TC-FX
N9K-	N9K-

C92160YC-X	C9348GC-FXP
N9K-C9272Q	N9K-X9732C-FX
N9K-C9236C	N9K-C9336C-FX2
N9K-C93240YC-FX2	N9K-C93300YC-FX2
N9K-C9364C	N9K-C9332C

Nexus Tahoe ASIC ELAM Procedure

Topology

Step 1 - Verify port's ASIC, Slice and SrcId

```
N9K-C92160YC-X-2# show hardware internal tah interface e1/49
IfIndex: 436232192
DstIndex: 5952
IfType: 26
Asic: 0 <<<<<<<<<
Asic: 0
AsicPort: 56
SrcId: 48 <<<<<<<<
Slice: 1 <<<<<<<<<
PortOnSlice: 24
```

> **Caution**: ELAM should be used only on one terminal window as you maintain global content for each slice, lu-a2d, etc.
> For example a port-channel (PO) may have two links, Eth 1/53 which corresponds to slice 0 and Eth 1/54 which corresponds to slice 1. Setting up ELAM on two separate terminal windows at a time for the different slices will not help as the latter slice (say slice 1) will overwrite the first one (slice 0), ending up in getting the same result on both terminal windows.

You can double check this information via:

```
N9K-C92160YC-X-2# show system internal ethpm info interface e1/49 | i i src
  IF_STATIC_INFO: port_name=Ethernet1/49,if_index:0x1a006000,ltl=5952,slot=0,
nxos_port=192,dmod=1,dpid=56,
  unit=0,queue=65535,xbar_unitbmp=0x0,ns_pid=255,slice_num=1,port_on_slice=24,src_id=48
```

Step 2 - Attach to the module

```
N9K-C92160YC-X-2# attach mod 1
```

Step 3 - Enter ELAM configuration mode and specify the proper ASIC from Step 1

```
module-1# debug platform internal tah elam asic 0
```

Step 4 - Configure the trigger - (There are many options you can specify here)

```
module-1(TAH-elam)# trigger init asic 0 slice 1 lu-a2d 1 in-select 6 out-select 0 use-src-id 48
```

Tip:

- If the ingress and egress ports are on different slices on the same ASIC, then ELAM on egress slice will not capture the outgoing packet because the packet will not go through the LUX Blocks on the egress slice and hence will bypass ELAM.
- lu-a2d 0 is used for reverse ELAM where the trigger is based on the result and lu-a2d 1 is used for ELAM where the trigger is based on packet attributes
- Use always 6 for in-select and 0 for out-select

Warning: Do not use 0 after lu-a2d as this might crash the switch - see CSCvd64106 for more details

Step 5 - Set the ELAM triggers using SRC & DEST IP from this example

```
module-1(TAH-elam-insel6)# reset
module-1(TAH-elam-insel6)# set outer ipv4 dst_ip 192.0.2.1 src_ip 192.0.2.2
```

Note: Make sure to "reset" as the "set" command will prevail throughout ELAMs and can cause it to not trigger or trigger on unexpected fields.

Step 6 - Start the ELAM

```
module-1(TAH-elam-insel6)# start
GBL_C++:  [MSG] tahusd_elam_wrapper_init:36:asic type 5 inst 0 slice 1 a_to_d 1 insel 6 outsel 0
GBL_C++:  [MSG] Inside tahusd_elam_wrapper_init
```

```
GBL_C++:  [MSG] tahusd_elam_wrapper_enable:7e:asic type 5 inst 0 slice 1 a_to_d 1
GBL_C++:  [MSG]   - writing
data=0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000001FD901C4E2A0000000000000000000020000000000000000000000000000000000000000000000
0000000000000000000030001
GBL_C++:  [MSG]   - writing
mask=0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000001FFFFFFFFFFFE000000000000000000000380000000000000000000000000000000000000000000000
000000000000000001FF001
```

Step 7 - Verify if your ELAM triggered and review the results

Note: Starting with NX-OS 7.0(3)I5(2) and later, there's a brief version of the ELAM report as seen below. If you are running pre-7.0(3)I5(2); continue to the "Viewing ELAM results..." section.

```
module-1(TAH-elam-insel6)# report

SUGARBOWL ELAM REPORT SUMMARY
slot - 1, asic - 1, slice - 1
============================

Incoming Interface: Eth1/49
Src Idx : 0xd, Src BD : 10
Outgoing Interface Info: dmod 1, dpid 14 Dst Idx : 0x602, Dst BD : 10 Packet Type: IPv4 Dst MAC
address: CC:46:D6:6E:28:DB
Src MAC address: 00:FE:C8:0E:27:15
.1q Tag0 VLAN: 10,  cos = 0x0

Dst IPv4 address: 192.0.2.1
Src IPv4 address: 192.0.2.2
Ver     =  4, DSCP    =    0, Don't Fragment = 0
Proto   =  1, TTL     =   64, More Fragments = 0
Hdr len = 20, Pkt len =   84, Checksum       = 0x667f

L4 Protocol  : 1
ICMP type    : 0
ICMP code    : 0

Drop Info:
----------

LUA:
LUB:
LUC:
LUD:
Final Drops:
```

"**`report detail`**" will provide you a more verbose output described in the later section and the default output on older releases.

The **Dst Idx** is the niv_idx of the egress interface. This can be checked with "**`show hardware internal tah interface ex/y | i niv`**"

The **dmod** and **dpid** are also corresponding to the egress interface. This can be checked with "**`show system internal ethpm info int ex/y | i dpid`**"

In addition, "show interface hardware-mapping" can be used to validate the dmod/dpid.

NOTES:

1. When it comes to confirming whether the packet is actually being dropped, the **"Final**

Drops" field is the ONLY one to consider. In other words, while you may see exceptions thrown in other fields like LUA/B/C/D, that *does not necessarily mean* that the packet is being dropped. Please review this output carefully.

2. Traffic punted to CPU will have the sup_hit flag set (**`report detail | grep sup_hit`**). You can decode the reason by using '**`show system internal access-list sup-redirect-stats all`**' and matching the sup index Ensure that the correct '**`system routing mode`**' is configured (show system routing mode) Per guidelines and limitations documented in Considerations for VXLAN Deployment The “System Routing Mode: template-vxlan-scale” is not applicable to Cisco NX-OS Release 7.0(3)I5(2) and later.When using VXLAN BGP EVPN in combination with Cisco NX-OS Release 7.0(3)I4(x) or NX-OS Release 7.0(3)I5(1), the “System Routing Mode: template-vxlan-scale” is required on the following hardware platforms:Cisco Nexus 9300-EX SwitchesCisco Nexus 9500 Switches with X9700-EX line cardsChanging the “System Routing Mode” requires a reload of the switch.

Example of traffic experiencing CPU punt:

```
module-1(TAH-elam-insel6)# report

SUGARBOWL ELAM REPORT SUMMARY
=============================

Incoming Interface: Eth1/3
Src Idx : 0x9, Src BD : 23
Outgoing Interface Info: dmod 1, dpid 72
Dst Idx : 0x601, Dst BD : 802

Packet Type: IPv4

Dst MAC address: B0:8B:CF:A3:D0:4B
Src MAC address: 00:10:DB:FF:10:00
.1q Tag0 VLAN: 23,  cos = 0x0

Dst IPv4 address: 192.0.2.1
Src IPv4 address: 192.0.2.2
Ver     =  4, DSCP    =    2, Don't Fragment = 1
Proto = 6, TTL = 49, More Fragments = 0 Hdr len = 20, Pkt len = 60, Checksum = 0x63c3 L4
Protocol : 6 TCP Dst Port : 80 TCP Src Port : 46340 Sup hit: 1, Sup Idx  : 2720  <<---- CPU
punt, use below CLI to resolve the meaning of Sup Idx

Drop Info:
----------

LUA:
LUB:
LUC:
LUD:
Final Drops:

# show system internal access-list sup-redirect-stats all | grep 2720

  2720                              copp-system-p-acl-http    63
```

Viewing ELAM results for NX-OS versions pre-7.0(3)I5(2)

+ Does this have a dot1q header?

```
module-1(TAH-elam-insel6)# report | grep pr_lu_vec_l2v.qtag0
GBL_C++:  [MSG]   pr_lu_vec_l2v.qtag0_vld: 0x1 << dot1q yes? 0x1
```

```
GBL_C++:  [MSG]   pr_lu_vec_l2v.qtag0_cos: 0x0
GBL_C++:  [MSG]   pr_lu_vec_l2v.qtag0_de: 0x0
GBL_C++:  [MSG]   pr_lu_vec_l2v.qtag0_vlan: 0xA << VL 10
```
+ Check VLAN:

```
module-1(TAH-elam-insel6)# report | grep -1 fpx_lookup_vec.lkup.macsakey.key.fid
GBL_C++:  [MSG]         fpx_lookup_vec.lkup.macsakey.key.vld: 0x1
GBL_C++:  [MSG]         fpx_lookup_vec.lkup.macsakey.key.fid_type: 0x0
GBL_C++:  [MSG]         fpx_lookup_vec.lkup.macsakey.key.fid_vld: 0x0
GBL_C++:  [MSG]         fpx_lookup_vec.lkup.macsakey.key.fid: 0xA << dec 0xa = VL 10
GBL_C++:  [MSG]         fpx_lookup_vec.lkup.macsakey.key.mac: 0xFEC80E2715
```
+ Check SRC MAC (you can actually see this in the previous step as well):

```
module-1(TAH-elam-insel6)# report | grep -i fpx_lookup_vec.lkup.macsakey.key.mac
GBL_C++:  [MSG]         fpx_lookup_vec.lkup.macsakey.key.mac: 0xFEC80E2715 << 00fe.c80e.2715
GBL_C++:  [MSG]         fpx_lookup_vec.lkup.macsakey.key.mac: 0xFEC80E2715
GBL_C++:  [MSG]         fpx_lookup_vec.lkup.macsakey.key.mac: 0xFEC80E2715
GBL_C++:  [MSG]         fpx_lookup_vec.lkup.macsakey.key.mac: 0xFEC80E2715
GBL_C++:  [MSG]         fpx_lookup_vec.lkup.macsakey.key.mac: 0xFEC80E2715
```
+ Is this a new learn?

```
module-1(TAH-elam-insel6)# report | grep -1 fpx_lookup_vec.sa_notify_info
GBL_C++:  [MSG]         fpx_lookup_vec.lkup.ptvec.misc1.tcp_flags: 0x0
GBL_C++:  [MSG]   fpx_lookup_vec.sa_notify_info: 0x5200000C060
GBL_C++:  [MSG]     fpx_lookup_vec.sa_notify_info.enable: 0x0 << This will be set to 0x1 for
learning to happen
GBL_C++:  [MSG]     fpx_lookup_vec.sa_notify_info.conv_learn_only: 0x0
<snip>
```
+ Check SRC & DST IP:

```
module-1(TAH-elam-insel6)# report | grep vec_l3v.ip.*a
GBL_C++:  [MSG]     pr_lu_vec_l3v.ip.da: 0x00000000000000000c0000201 << DST IP: 192.0.2.1
GBL_C++:  [MSG]     pr_lu_vec_l3v.ip.sa: 0x00000000000000000c0000202 << SRC IP: 192.0.2.2
```

+ Verify your ingress SRC_ID:

```
module-1(TAH-elam-insel6)# report | egrep SRC
GBL_C++:  [MSG]    SRCID: 0x30

module-1(TAH-elam-insel6)# report | grep vec.ihdr.ieth.hdr.src_idx
GBL_C++: [MSG] lurw_vec.ihdr.ieth.hdr.src_idx: 0xA9 << sh hardware internal tah int e1/49 | i i
niv_idx
```

+ If ELAM does not trigger, it will look as follows:

```
module-1(TAH-elam-insel6)# report
GBL_C++:  [MSG] tahusd_elam_wrapper_report:27d:asic type 5 inst 0 slice 1 a_to_d 1 insel 6
outsel 0
GBL_C++:  [MSG] Inside tahusd_elam_wrapper_dav_report
GBL_C++:  [MSG] ELAM not yet triggered <<<<<<
```

VXLAN Encapsulated ELAM:

Since VXLAN packets would be encapsulated, the ELAM needs to be triggered on the INNER header as opposed to the OUTER header - See example below for an ARP frame:

```
module-1# debug platform internal tah elam asic 0
module-1(TAH-elam)# trigger init asic 0 slice 1 in-select 7 out-select 0 use-src-id 48
module-1(TAH-elam-insel7)# reset
module-1(TAH-elam-insel7)# set inner arp source-ip-addr 192.0.2.2 target-ip-addr 192.0.2.1
module-1(TAH-elam-insel7)# start
module-1(TAH-elam-insel7)# report
```

ELAM configuration post NX-OS 7.0(3)I7(2)

Post NX-OS 7.0(3)I7(2), ELAM can now be triggered globally without specifying the ASIC or Slice number for ease - See example below:

```
Nexus-9K# debug platform internal tah elam
Nexus-9K(TAH-elam)# trigger init
Nexus-9K(TAH-elam-insel6)# reset
Nexus-9K(TAH-elam-insel6)# set outer ipv4 dst_ip 192.0.2.1 src_ip 192.0.2.2
Nexus-9K(TAH-elam-insel6)# start
Nexus-9K(TAH-elam-insel6)# report
```