EX硬件：ACI信息包转发深潜。

目录

简介

本文描述不同的转发方案使用在应用程序中心基础设施(ACI)的新一代ASIC硬件。 它将显示如何验证硬件正确地被编程，并且我们是转发数据包对正确目的地终端(EP)在适合的终端组(EPGs)中。

先决条件

要求

本文档没有任何特定的要求。

使用的组件

本文档中的信息基于下列硬件和软件版本：

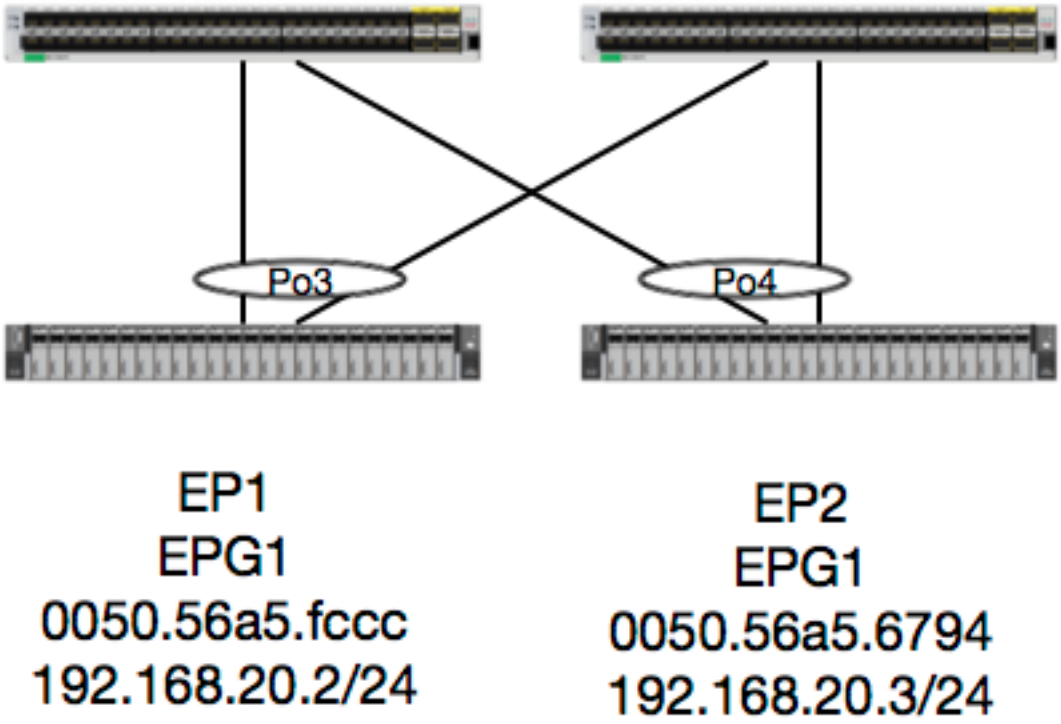
- 使用EX硬件，包括两脊椎交换机和两分支交换机的ACI结构
- 有去其中每一分支的两uplink端口的一台ESXi主机交换
- 作为路由器的连结5000设备。
- 使用初始设置的应用程序策略基础设施控制器(APIC)

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您使用的是真实网络，请确保您已经了解所有命令的潜在影响。

方案

在同样EPG/Same分支的2个EP -交换式帧

拓扑



给此拓扑，流从EP1到EP2是L2流并且应该交换本地在任何分支源数据流进来在。 检查的第一件事与Layer2 (L2)流是确定的MAC地址表是否和交换机收到的帧的地方：

```
leaf4# show mac address-table | grep fccc * 30 0050.56a5.fccc dynamic - F F po3 leaf4# show mac address-table | grep 6794 * 30 0050.56a5.6794 dynamic - F F po4
```

为了看到封装VLAN，我们能检查EP数据库：

```
leaf4# show endpoint mac 0050.56a5.fccc Legend: O - peer-attached H - vtep a - locally-aged S - static V - vpc-attached p - peer-aged L - local M - span s - static-arp B - bounce +------------------------+--------------+----------------+-------------+-------------+ VLAN/ Encap MAC Address MAC Info/ Interface Domain VLAN IP Address IP Info +-------------------------+--------------+----------------+-------------+-------------+ 30 vlan-2268 0050.56a5.fccc LV po3 Joey-Tenant:Joey-Internal vlan-2268 192.168.20.2 LV po3 calo2-leaf4# show endpoint mac 0050.56a5.6794 Legend: O - peer-attached H - vtep a - locally-aged S - static V - vpc-attached p - peer-aged L - local M - span s - static-arp B - bounce +------------------------+--------------+----------------+-------------+-------------+ VLAN/ Encap MAC Address MAC Info/ Interface Domain VLAN IP Address IP Info +---------------------------------+--------------+----------------+-------------+-------------+ 30 vlan-2268 0050.56a5.6794 LV po4 Joey-Tenant:Joey-Internal vlan-2268 192.168.20.3 LV po4
```

我们在软件方面认识FD_VLAN 30匹配，但是我们能总是验证映射：

```
leaf4# show vlan extended | grep 2268 30 enet CE vlan-2268
```

当然并且，我们能检查硬件确保VLAN 30地图到VLAN 2268作为前面板封装。

```
leaf4# vsh_lc
module-1# show system internal eltmc info vlan 30 vlan_id: 30 ::: hw_vlan_id: 22 vlan_type:
FD_VLAN ::: bd_vlan: 28 access_encap_type: 802.1q ::: access_encap: 2268 fabric_encap_type:
VXLAN ::: fabric_encap: 11960 sclass: 32778 ::: scope: 11 untagged: 0 acess_encap_hex: 0x8dc :::
fabric_enc_hex: 0x2eb8 pd_vlan_ft_mask: 0x8 fd_learn_disable: 0 qos_class_id: 0 ::: qos_pap_id:
0 qq_met_ptr: 25 ::: ipmc_index: 0 ingressBdAclLabel: 0 ::: ingBdAclLblMask: 0 egressBdAclLabel:
0 ::: egrBdAclLblMask: 0 qos_map_idx: 0 ::: qos_map_pri: 0 qos_map_dscp: 0 ::: qos_map_tc: 0
vlan_ft_mask: 0xe30 hw_bd_idx: 0 ::: hw_epg_idx: 11267 intf_count: 2 ::: glbl_scp_if_cnt: 2
<SNIPPED>
```

在的情况下EP在软件方面学习，我们能也验证硬件编程了这些EP的L2信息。 在新的硬件中，有是硬件的软件状态的硬件抽象层(HAL)。 HAL的工作是采取编程请求和推送他们对硬件。

为了查看L2关于终端的硬件信息，我们能查看在HAL的L2表为给的MAC地址：

```
leaf4# vsh_lc
module-1# show platform internal hal ep l2 mac 0050.56a5.fccc LEGEND: ------- BDId: BD Id BD
Name: BD Name T: EP Type (Pl: Physical Vl: Virtual Xr: Remote EP Mac: Mac L2 IfId: L2 Interface
L2 IfName: L2 IfName FDId: FD Id FD Name: FD Name S Class: S Class Age Intvl: Age Interval P A:
Packet Action (F: Forward, T: Trap to CPU, L: Log & Forward, D: Drop, N: None) S T: Static Ep S
E: Secure EP L D: Learn Disable B N D: Bind Notify Disable E N D: Epg Notify Disable B E: Bounce
Enable I D L: IVxlan Dont Learn SPI: Source Policy Incomplete DPI: Dest Policy Incomplete SPA:
Source Policy Applied DPA: Dest Policy Applied DSS: Dest Shared Service IL: Is Local VUB: Vnid
Use Bd SO: SA Only L2 EP Count: 1
================================================================================================
====================== B E I S D S D D V BD EP L2 L2 FD S Age P S S L N N B D P P P P S I U S
BdId Name T Mac IfId Ifname FDId Name Class Intvl A T E D D D E L I I A A S L B O
================================================================================================
====================== 1c BD-28 Pl 00:50:56:a5:fc:cc 16000002 Po3 1e FD-30 800a 29f F 0 0 0 1 0
0 0 0 0 0 0 0 1 0 0 module-1# show platform internal hal ep l2 mac 0050.56a5.6794
================================================================================================
====================== B E I S D S D D V BD EP L2 L2 FD S Age P S S L N N B D P P P P S I U S
BdId Name T Mac IfId Ifname FDId Name Class Intvl A T E D D D E L I I A A S L B O
================================================================================================
====================== 1c BD-28 Pl 00:50:56:a5:67:94 16000003 Po4 1e FD-30 800a 29f F 0 0 0 1 0
0 0 0 0 0 0 0 1 0 0
```

即然我们映射硬件，请执行伊拉姆和发现数据包哪里应该是。

伊拉姆

```
leaf4# vsh_lc
module-1# debug platform internal tah elam asic 0 module-1(DBG-TAH-elam)# trigger reset module-
1(DBG-TAH-elam)# trigger init in-select 6 out-select 0 module-1(DBG-TAH-elam-insel6)# set outer
l2 src_mac 0050.56a5.fccc dst_mac 0050.56a5.6794 module-1(DBG-TAH-elam-insel6)# start module-
1(DBG-TAH-elam-insel6)# stat ELAM STATUS ========== Asic 0 Slice 0 Status Armed Asic 0 Slice 1
Status Triggered module-1(DBG-TAH-elam-insel6)# report | grep ovec
sug_elam_out_sidebnd_no_spare_vec.ovector_idx: 0x9E
```

极大， Leaf4如此接收在Asic的帧0片式1。 使用是非常重要的新的硬件的伊拉姆，有新字段，当排除故障时：ovector_idx。 此索引是物理端口索引应该转发帧/数据包在外面。 一旦有ovector_idx，我们能使用此命令查找什么端口映射到：

```
module-1(DBG-TAH-elam-insel6)# show platform internal hal l2 port gpd Legend: ------- IfId:
Interface Id IfName: Interface Name I P: Is PC Mbr IfId: Interface Id Uc PC Cfg: UcPcCfg Idx Uc
PC MbrId: Uc Pc Mbr Id As: Asic AP: Asic Port Sl: Slice Sp: Slice Port Ss: Slice SrcId Ovec:
Ovector (slice | srcid) L S: Local Slot Reprogram: L3: Is L3 P: PifTable Xla Idx: Xlate Idx RP:
```

```
Rw PifTable Ovx Idx: OXlate Idx IP: If Profile Table N L3: Num. of L3 Ifs RS: Rw SrcId Table NI
L3: Num. of Infra L3 Ifs DP: DPort Table Vif Tid: Vif Tid SP: SrcPortState Table RwV Tid: RwVif
Tid RSP: RwSrcPortstate Table Ing Lbl: Ingress Acl Label UC: UCPcCfg Egr Lbl: Egress Acl Label
UM: UCPcMbr Reprogram: PROF ID: Lport Profile Id VS: VifStateTable HI: LportProfile Hw Install
RV: Rw VifTable Num. of Sandboxes: 1 Sandbox_ID: 0, BMP: 0x0 Port Count: 8
================================================================================================
========================================== Uc Uc | Reprogram | | Rep | I PC Pc L | R I R D R U
U X | L Xla Ovx N NI Vif RwV Ing Egr | V R | PROF H IfId Ifname P Cfg MbrID As AP Sl Sp Ss Ovec
S | P P P S P Sp Sp C M L | 3 Idx Idx L3 L3 Tid Tid Lbl Lbl | S V | ID I
================================================================================================
========================================== 1a004000 Eth1/5 1 0 1d 0 d 0 c 18 18 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 - - 800 0 0 1 0 0 1a005000 Eth1/6 1 0 b 0 e 0 d 1a 1a 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 - - 800 0 0 1 0 0 1a006000 Eth1/7 0 26 5 0 f 0 e 1c 1c 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
D-256 - 800 0 0 1 e 0 1a007000 Eth1/8 0 2e 7 0 10 0 f 1e 1e 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 D-84
- 800 0 0 1 30 0 1a01e000 Eth1/31 1 0 2d 0 37 1 e 1c 9c 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - - 0 0
0 1 0 0 1a01f000 Eth1/32 1 0 3d 0 38 1 f 1e 9e 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - - 0 0 0 1 0 0
1a030000 Eth1/49 0 2 1 0 49 1 20 38 b8 1 0 0 0 0 0 0 0 0 0 0 1 8 6 2 2 D-24d - 400 0 0 0 1 0
1a031000 Eth1/50 0 3 3 0 29 1 0 0 80 1 0 0 0 0 0 0 0 0 0 0 1 9 7 2 2 D-350 - 400 0 0 0 1 0
```

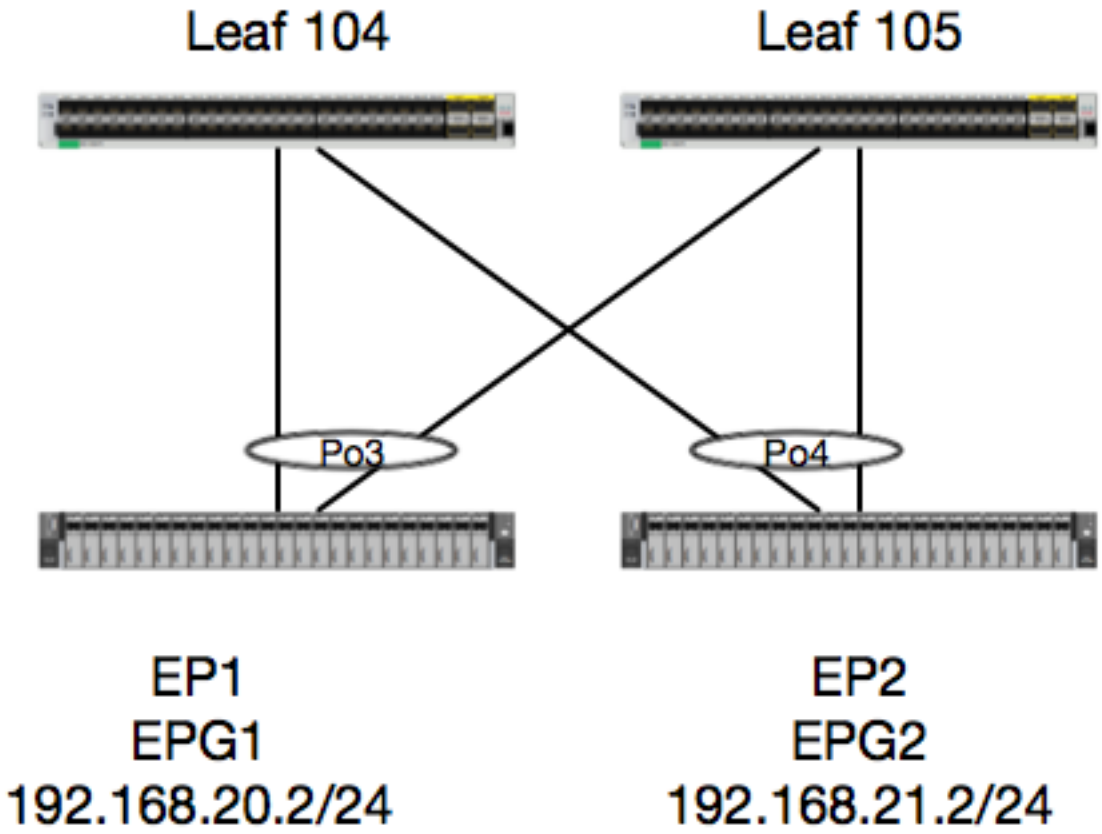
交换机认为应该转发数据包在以太网接口1/32外面。 该PO4我们了解mac addess的地方？

```
leaf4# show port-channel summary
Flags:  D - Down        P - Up in port-channel (members)
        I - Individual  H - Hot-standby (LACP only)
        s - Suspended   r - Module-removed
        S - Switched    R - Routed
        U - Up (port-channel)
        M - Not in use. Min-links not met
        F - Configuration failed
------------------------------------------------------------------------------
Group Port-       Type     Protocol  Member Ports
      Channel
------------------------------------------------------------------------------
1     Po1(SU)     Eth      LACP      Eth1/5(P)
2     Po2(SU)     Eth      LACP      Eth1/6(P)
3     Po3(SU)     Eth      LACP      Eth1/31(P)
4 Po4(SU) Eth LACP Eth1/32(P)
```

是，因此数据包增殖比转发在接口1/32外面对目的地主机。

在不同的EPG/Same分支的2个EP -路由信息包

拓扑



在本例中，我们将跟踪一数据包的数据包流从EP1到他们在同一个vPC分支对存在的EP2。 使用不同的BD'S，两个EP是用不同的EPG。

总是要执行的第一件事是检查EP数据库发现我们是否了解EP ：

```
leaf4# show endpoint ip 192.168.20.2 Legend: O - peer-attached H - vtep a - locally-aged S -
static V - vpc-attached p - peer-aged L - local M - span s - static-arp B - bounce +------------
----------------------+--------------+----------------+-------------+-------------+ VLAN/
Encap MAC Address MAC Info/ Interface Domain VLAN IP Address IP Info +--------------------------
---------+--------------+----------------+-------------+-------------+ 30 vlan-2268
0050.56a5.fccc LV po3 Joey-Tenant:Joey-Internal vlan-2268 192.168.20.2 LV po3 calo2-leaf4# show
endpoint ip 192.168.21.2 Legend: O - peer-attached H - vtep a - locally-aged S - static V - vpc-
attached p - peer-aged L - local M - span s - static-arp B - bounce +--------------------------
--------+--------------+----------------+-------------+-------------+ VLAN/ Encap MAC Address
MAC Info/ Interface Domain VLAN IP Address IP Info +---------------------------------+--------
-------+----------------+-------------+-------------+ 8 vlan-2200 0050.56a5.0c11 LV po4 Joey-
Tenant:Joey-Internal vlan-2200 192.168.21.2 LV po4
```

因为我们了解EP并且认识IP信息，我们在硬件方面的应该能查看EP学习信息：

```
leaf4# vsh_lc module-1# show platform internal hal ep l3 all LEGEND: ------- VrfName: Vrf Name
T: Type (Pl: Physical, Vl: Virtual, Xr: Remote) EP IP: Endpoint IP S Class: S Class Age Intvl:
Age Interval S T: Static Ep S E: Secure EP L D: Learn Disable B N D: Bind Notify Disable E N D:
Epg Notify Disable B E: Bounce Enable I D L: IVxlan Dont Learn SPI: Source Policy Incomplete
DPI: Dest Policy Incomplete SPA: Source Policy Applied DPA: Dest Policy Applied DSS: Dest Shared
Service IL: Is Local VUB: Vnid Use Bd SO: SA Only EP NH L3IfName: EP Next Hop L3 If Name NHT:
Next Hop Type (L2: L2 Entry L3: L3 Next Hop) BD Name: L2 NH BD Name EP Mac: EP Mac L3 IfName: L3
NH If Name L2 IfName: L2 If Name FD Name: L2 Entry FD Name IP: L3 NH IP L3 EP Count: 12
================================================================================================
================================================================================ B E
I S D S D D V EP-NH N | Vrf EP S Age S S L N N B D P P P P S I U S L3 H | BD EP L3 L2 FD Name T
IP Class Intvl T E D D D E L I I A A S L B O IfName T | Name Mac IfName Ifname Name IP
================================================================================================
================================================================================
```

```
common*rewall Pl 10.6.112.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - -
0.0.0.0 common*rewall Pl 10.6.114.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 -
- - 0.0.0.0 common*rewall Pl 10.6.114.129 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 -
00:00:00:00:00:00 - - - 0.0.0.0 common*efault Pl 100.100.101.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0
- L3 - 00:00:00:00:00:00 - - - 0.0.0.0 Joey-T*ternal Pl 192.168.1.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0
1 0 0 - L3 - 00:00:00:00:00:00 - - - 0.0.0.0 Joey-T*ternal Xr 192.168.1.100 8013 128 0 0 0 1 0 0
0 0 0 0 0 0 0 1 0 - L3 - 00:0c:0c:0c:0c:0c Tunnel2 Tunnel2 - 0.0.0.0 Joey-T*ernal2 Pl
192.168.3.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - - 0.0.0.0 Joey-
T*ternal Pl 192.168.20.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - -
0.0.0.0 Joey-T*ternal Pl 192.168.20.2 800a 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 - L2 BD-28
00:50:56:a5:fc:cc - Po3 FD-30 - Joey-T*ternal Pl 192.168.21.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0
- L3 - 00:00:00:00:00:00 - - - 0.0.0.0 Joey-T*ternal Pl 192.168.21.2 800c 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 - L2 BD-7 00:50:56:a5:0c:11 - Po4 FD-8 - Joey-T*ternal Pl 2001:0:0:100::1 1 0 1 0 0 0
0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - - 0.0.0.0
```

HAL第3层(l3)表是非常uself，因为提供我们l3了解的EP的VLAN/Port信息。 我们知道目的地存在Po4，因此应该转发数据包在Po4的所有端口外面。

请运行伊拉姆和发现什么我们获得!

伊拉姆

```
leaf4# vsh_lc
module-1# debug platform internal tah elam asic 0 module-1(DBG-TAH-elam)# trigger init in-select
6 out-select 0 module-1(DBG-TAH-elam-insel6)# set outer ipv4 src_ip 192.168.20.2 dst_ip
192.168.21.2 module-1(DBG-TAH-elam-insel6)# start module-1(DBG-TAH-elam-insel6)# stat ELAM
STATUS ========== Asic 0 Slice 0 Status Armed Asic 0 Slice 1 Status Armed module-1(DBG-TAH-
elam-insel6)# stat ELAM STATUS ========== Asic 0 Slice 0 Status Armed Asic 0 Slice 1 Status
Triggered module-1(DBG-TAH-elam-insel6)# report | grep ovec
sug_elam_out_sidebnd_no_spare_vec.ovector_idx: 0x9E
```

极大，因此我们触发了数据包和我们发现“ovector_idx”是0x9E。 ovector索引是流出的phycial接口索引应该转发数据包在外面。 请发现什么端口有该索引：

```
module-1(DBG-TAH-elam-insel6)# show platform internal hal l2 port gpd Legend: ------- IfId:
Interface Id IfName: Interface Name I P: Is PC Mbr IfId: Interface Id Uc PC Cfg: UcPcCfg Idx Uc
PC MbrId: Uc Pc Mbr Id As: Asic AP: Asic Port Sl: Slice Sp: Slice Port Ss: Slice SrcId Ovec:
Ovector (slice | srcid) L S: Local Slot Reprogram: L3: Is L3 P: PifTable Xla Idx: Xlate Idx RP:
Rw PifTable Ovx Idx: OXlate Idx IP: If Profile Table N L3: Num. of L3 Ifs RS: Rw SrcId Table NI
L3: Num. of Infra L3 Ifs DP: DPort Table Vif Tid: Vif Tid SP: SrcPortState Table RwV Tid: RwVif
Tid RSP: RwSrcPortstate Table Ing Lbl: Ingress Acl Label UC: UCPcCfg Egr Lbl: Egress Acl Label
UM: UCPcMbr Reprogram: PROF ID: Lport Profile Id VS: VifStateTable HI: LportProfile Hw Install
RV: Rw VifTable Num. of Sandboxes: 1 Sandbox_ID: 0, BMP: 0x0 Port Count: 8
================================================================================================
========================================== Uc Uc | Reprogram | | Rep | I PC Pc L | R I R D R U
U X | L Xla Ovx N NI Vif RwV Ing Egr | V R | PROF H IfId Ifname P Cfg MbrID As AP Sl Sp Ss Ovec
S | P P P S P Sp Sp C M L | 3 Idx Idx L3 L3 Tid Tid Lbl Lbl | S V | ID I
================================================================================================
========================================== 1a004000 Eth1/5 1 0 1d 0 d 0 c 18 18 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 - - 800 0 0 1 0 0 1a005000 Eth1/6 1 0 b 0 e 0 d 1a 1a 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 - - 800 0 0 1 0 0 1a006000 Eth1/7 0 26 5 0 f 0 e 1c 1c 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
D-256 - 800 0 0 1 c 0 1a007000 Eth1/8 0 2f 7 0 10 0 f 1e 1e 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 D-
199 - 800 0 0 1 2e 0 1a01e000 Eth1/31 1 0 2d 0 37 1 e 1c 9c 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - -
0 0 0 1 0 0 1a01f000 Eth1/32 1 0 3d 0 38 1 f 1e 9e 1 0 0 0 0 0 0 0 0 0 0 0 0 0 - - 0 0 0 1 0
0 1a030000 Eth1/49 0 2 1 0 49 1 20 38 b8 1 0 0 0 0 0 0 0 0 0 0 1 6 4 2 2 D-24d - 400 0 0 0 1 0
1a031000 Eth1/50 0 3 3 0 29 1 0 0 80 1 0 0 0 0 0 0 0 0 0 0 1 5 3 2 2 D-350 - 400 0 0 0 1 0
```

查找，如我们应该传送它波尔特1/32，是更正？

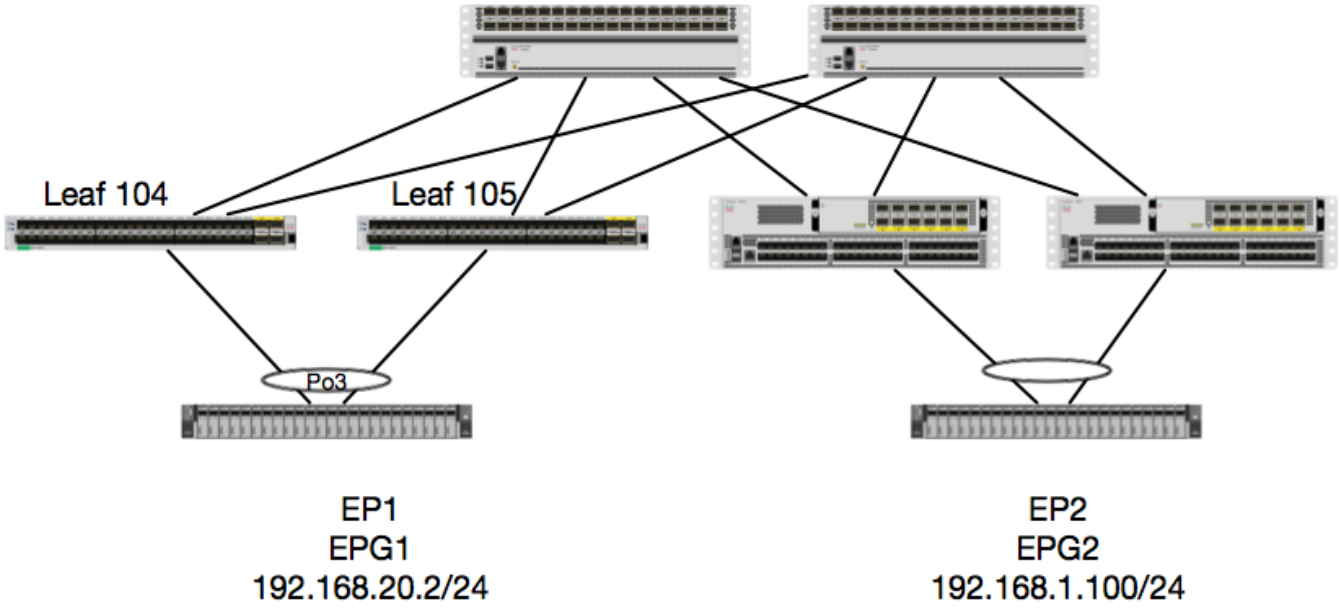
```
leaf4# show port-channel summary
Flags:  D - Down        P - Up in port-channel (members)
        I - Individual  H - Hot-standby (LACP only)
        s - Suspended   r - Module-removed
```

```
        S - Switched    R - Routed
        U - Up (port-channel)
        M - Not in use. Min-links not met
        F - Configuration failed
------------------------------------------------------------------------------
Group Port-       Type     Protocol  Member Ports
      Channel
------------------------------------------------------------------------------
1     Po1(SU)     Eth      LACP      Eth1/5(P)
2     Po2(SU)     Eth      LACP      Eth1/6(P)
3     Po3(SU)     Eth      LACP      Eth1/31(P)
4 Po4(SU) Eth LACP Eth1/32(P)
```

是，这正确。

在不同的EPG/Different分支的2个EP -路由信息包

拓扑



在本例中，我们将跟踪一数据包的数据包流从EP1到EP1在一个EX vPC对存在的EP2，并且EP2在一个远程生成1 vPC分支对存在。 使用不同的BD'S，两个EP是用不同的EPG。

再次，请检查EP哪里学习：

```
leaf4# show endpoint ip 192.168.20.2 Legend: O - peer-attached H - vtep a - locally-aged S -
static V - vpc-attached p - peer-aged L - local M - span s - static-arp B - bounce +------------
----------------------+---------------+-----------------+-------------+-------------+ VLAN/
Encap MAC Address MAC Info/ Interface Domain VLAN IP Address IP Info +--------------------------
---------+---------------+-----------------+-------------+-------------+ 30 vlan-2268
0050.56a5.fccc LV po3 Joey-Tenant:Joey-Internal vlan-2268 192.168.20.2 LV po3 calo2-leaf4# show
endpoint ip 192.168.1.100 Legend: O - peer-attached H - vtep a - locally-aged S - static V -
vpc-attached p - peer-aged L - local M - span s - static-arp B - bounce +----------------------
------------+---------------+-----------------+-------------+-------------+ VLAN/ Encap MAC
Address MAC Info/ Interface Domain VLAN IP Address IP Info +---------------------------------
+---------------+-----------------+-------------+-------------+ Joey-Tenant:Joey-Internal
192.168.1.100 tunnel2
```

现在，请验证什么硬件编程了：

```
leaf4# vsh_lc
```

module-1# **show platform internal hal ep l3 all** LEGEND: ------- VrfName: Vrf Name T: Type (Pl: Physical, Vl: Virtual, Xr: Remote) EP IP: Endpoint IP S Class: S Class Age Intvl: Age Interval S T: Static Ep S E: Secure EP L D: Learn Disable B N D: Bind Notify Disable E N D: Epg Notify Disable B E: Bounce Enable I D L: IVxlan Dont Learn SPI: Source Policy Incomplete DPI: Dest Policy Incomplete SPA: Source Policy Applied DPA: Dest Policy Applied DSS: Dest Shared Service IL: Is Local VUB: Vnid Use Bd SO: SA Only EP NH L3IfName: EP Next Hop L3 If Name NHT: Next Hop Type (L2: L2 Entry L3: L3 Next Hop) BD Name: L2 NH BD Name EP Mac: EP Mac L3 IfName: L3 NH If Name L2 IfName: L2 If Name FD Name: L2 Entry FD Name IP: L3 NH IP L3 EP Count: 12
== B E I S D S D D V EP-NH N | Vrf EP S Age S S L N N B D P P P P S I U S L3 H | BD EP L3 L2 FD Name T IP Class Intvl T E D D D E L I I A A S L B O IfName T | Name Mac IfName Ifname Name IP
==
common*rewall Pl 10.6.112.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - - 0.0.0.0 common*rewall Pl 10.6.114.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - - 0.0.0.0 common*rewall Pl 10.6.114.129 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - - 0.0.0.0 common*efault Pl 100.100.101.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - - 0.0.0.0 Joey-T*ternal Pl 192.168.1.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - - 0.0.0.0 **Joey-T*ternal Xr 192.168.1.100 8013 128 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 - L3 - 00:0c:0c:0c:0c:0c Tunnel2 Tunnel2 - 0.0.0.0** Joey-T*ernal2 Pl 192.168.3.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - - 0.0.0.0 Joey-T*ternal Pl 192.168.20.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - - 0.0.0.0 **Joey-T*ternal Pl 192.168.20.2 800a 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 - L2 BD-28 00:50:56:a5:fc:cc - Po3 FD-30 -** Joey-T*ternal Pl 192.168.21.1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - - 0.0.0.0 Joey-T*ternal Pl 192.168.21.2 800c 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 - L2 BD-7 00:50:56:a5:0c:11 - Po4 FD-8 - Joey-T*ternal Pl 2001:0:0:100::1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 - L3 - 00:00:00:00:00:00 - - - 0.0.0.0

硬件认为EP在通道2.存在。 什么是通道的2目的地？

module-1# **show system internal eltmc info interface tunnel2** IfInfo: interface: **Tunnel2** ::: ifindex: **402718722** iod: 66 ::: state: up Mod: 0 ::: Port: 0 Tunnel Index: 0 ::: Tunnel Dst ip: 0xc0a87843 Tunnel Encap: ivxlan ::: Tunnel VPC Peer: 0 Tunnel Dst ip str: **192.168.120.67** ::: Tunnel ept: 0x1 [SDK Info]: tunnl_name: vrf_id: 2 ::: if_index: 0x18010002 hwencapidx: 0 ::: encaptype: 1 mac_proxy: 0 ::: v4_proxy: 0 v6_proxy: 0 ::: ip_addr_type: 0 ipv4_address: 0xc0a87843 [SDB INFO]: iod: 66 pc_if_index: 0 fab_if_index: 0 sv_if: 0 src_idx: 0 int_vlan: 0 encap_vlan: 0 mod_port_status: 0x41620003 v6_tbl_id: 0x80000002 v4_tbl_id: 0x2 router_mac:00.00.00.00.00.00 unnumbered: 0 trunk_id: 0 tunnel_mod: 0 tunnel_port: 0 tep_ip: 0xc0a87843 ip_if_mode: 0 sdk_vrf_id: 2 mtu: 9366 ::: ipmtu_id: 0 is_fex_fabric: 0

因为目的地存在vPC，该目的地IP应该是远程分支的vPC虚拟IP。 请检查一个远程分支和看：

leaf1# **show system internal epm vpc** Local TEP IP : 192.168.160.95 Peer TEP IP : 192.168.160.93 vPC configured : Yes **vPC VIP : 192.168.120.67** MCT link status : Up Local vPC version bitmap : 0x7 Peer vPC version bitmap : 0x7 Negotiated vPC version : 3 Peer advertisement received : Yes Tunnel to vPC peer : Up

完善，因此它学习从远程vPC对的目的地EP。 请发现什么伊拉姆看到，并且请验证我们转发数据包correectly ：

伊拉姆

module-1# debug platform internal tah elam asic 0
module-1(DBG-TAH-elam)# **trigger init in-select 6 out-select 0** module-1(DBG-TAH-elam-insel6)# **set outer ipv4 src_ip 192.168.20.2 dst_ip 192.168.1.100** module-1(DBG-TAH-elam-insel6)# start module-1(DBG-TAH-elam-insel6)# stat ELAM STATUS ========== Asic 0 Slice 0 Status Armed **Asic 0 Slice 1 Status Triggered**

现在，与在EX硬件的远程目的地，有是非常重要，当排除故障数据包流时的2个伊拉姆值。 ovector_idx喜欢前面和encap_idx ：

module-1(DBG-TAH-elam-insel6)# **report | grep ovec** sug_elam_out_sidebnd_no_spare_vec.ovector_idx: **0xB8** module-1(DBG-TAH-elam-insel6)# **report | grep encap** sug_lurw_vec.encap_l2_idx: 0x0

```
sug_lurw_vec.encap_pcid: 0x0 sug_lurw_vec.encap_idx: 0x6 sug_lurw_vec.encap_vld: 0x1
```

在EX硬件上，我们有能力驱动应该转发在外面数据包的目的地端口。 前面，我们通常检查encap idx并且验证目的地idx是正确通道。 此处我们能验证对8B的什么端口映射：

```
module-1(DBG-TAH-elam-insel6)# show platform internal hal l2 port gpd Legend: ------- IfId:
Interface Id IfName: Interface Name I P: Is PC Mbr IfId: Interface Id Uc PC Cfg: UcPcCfg Idx Uc
PC MbrId: Uc Pc Mbr Id As: Asic AP: Asic Port Sl: Slice Sp: Slice Port Ss: Slice SrcId Ovec:
Ovector (slice | srcid) L S: Local Slot Reprogram: L3: Is L3 P: PifTable Xla Idx: Xlate Idx RP:
Rw PifTable Ovx Idx: OXlate Idx IP: If Profile Table N L3: Num. of L3 Ifs RS: Rw SrcId Table NI
L3: Num. of Infra L3 Ifs DP: DPort Table Vif Tid: Vif Tid SP: SrcPortState Table RwV Tid: RwVif
Tid RSP: RwSrcPortstate Table Ing Lbl: Ingress Acl Label UC: UCPcCfg Egr Lbl: Egress Acl Label
UM: UCPcMbr Reprogram: PROF ID: Lport Profile Id VS: VifStateTable HI: LportProfile Hw Install
RV: Rw VifTable Num. of Sandboxes: 1 Sandbox_ID: 0, BMP: 0x0 Port Count: 8
================================================================================================
========================================== Uc Uc | Reprogram | | Rep | I PC Pc L | R I R D R U
U X | L Xla Ovx N NI Vif RwV Ing Egr | V R | PROF H IfId Ifname P Cfg MbrID As AP Sl Sp Ss Ovec
S | P P P S P Sp Sp C M L | 3 Idx Idx L3 L3 Tid Tid Lbl Lbl | S V | ID I
================================================================================================
========================================== 1a004000 Eth1/5 1 0 1d 0 d 0 c 18 18 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 - - 800 0 0 1 0 0 1a005000 Eth1/6 1 0 b 0 e 0 d 1a 1a 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 - - 800 0 0 1 0 0 1a006000 Eth1/7 0 26 5 0 f 0 e 1c 1c 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
D-256 - 800 0 0 1 c 0 1a007000 Eth1/8 0 2f 7 0 10 0 f 1e 1e 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 D-
199 - 800 0 0 1 2e 0 1a01e000 Eth1/31 1 0 2d 0 37 1 e 1c 9c 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - -
0 0 0 1 0 0 1a01f000 Eth1/32 1 0 3d 0 38 1 f 1e 9e 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - - 0 0 0 1 0
0 1a030000 Eth1/49 0 2 1 0 49 1 20 38 b8 1 0 0 0 0 0 0 0 0 0 1 6 4 2 2 D-24d - 400 0 0 0 1 0
1a031000 Eth1/50 0 3 3 0 29 1 0 0 80 1 0 0 0 0 0 0 0 0 0 1 5 3 2 2 D-350 - 400 0 0 0 1 0
```

交换机认为应该转发它到在接口Eth1/49的脊椎。 但是如何能验证encap是正确？

我们首先需要查看关于通道的硬件信息。 我们能通过运行此HAL命令执行此：

```
module-1(DBG-TAH-elam-insel6)# show platform internal hal tunnel rtep pi Non-Sandbox Mode
LEGEND: ------- Tun Ifid: Tunnel Ifid IfName: Tunnel If Name Lid: Logical Id ET: Encap Type V:
Vxlan I: IVxlan N: NVGRE VrfId: Vrf Id Vrf Name: Vrf Name IP: Tunnel's IP Hw Enc: Hw Encap Idx
IVP: Is VPC Peer IL: Is Local P4: Proxy for v4 P6: Proxy for V6 PM: Proxy for Mac II: Is Ingress
Only IC: Is Copy Service C OBd: Copy Service Outer Bd U D: Use DF NBT: Next Base Type E: ECMP N:
Next-Hop NB Id: Next Base Id NH cnt: Next Hop Count VrfId: Vrf Id Vrf Name: Vrf Name IP: IP
Address Mac: Mac L3 IfId: L3 IfId L3IfName: L3 If Name L2 IfId: L2 IfId L2IfName: L2 If Name
Num. of Sandboxes: 1 Sandbox_ID: 0, BMP: 0x0 Remote Tep Count: 15
================================================================================================
================================================================================================
======= I N N | E Vrf Hw V I P P P I I C U B B NH | Vrf L3 L3 L2 L2 IfId Ifname T Lid VrfId Name
IP Enc P L 4 6 M I C OBd D T Id Cnt | VrfId Name IP Mac IfId IfName IfId IfName
================================================================================================
================================================================================================
======= 18010002 Tunnel2 I 3005 2 overlay-1 192.168.120.670 0 0 0 0 0 0 0 1 0 E 2 2 2 overlay-1
0.0.0.0 0d:0d:0d:0d:0d:00 1a030001 Eth1/49.1 1a030000 Eth1/4 9 2 overlay-1 0.0.0.0
0d:0d:0d:0d:0d:00 1a031002 Eth1/50.2 1a031000 Eth1/5 0
```

此输出给我们我们关心的一些个值：

IfId -接口ID分配到通道

IP -目的地的IP。 这应该匹配ELTMC。

L3 IfId -交换机能使用转发到适当的目的地的第3层接口。

一旦我们认识IfId，我们能验证我们在elam获得匹配隧道目的地的encap ：

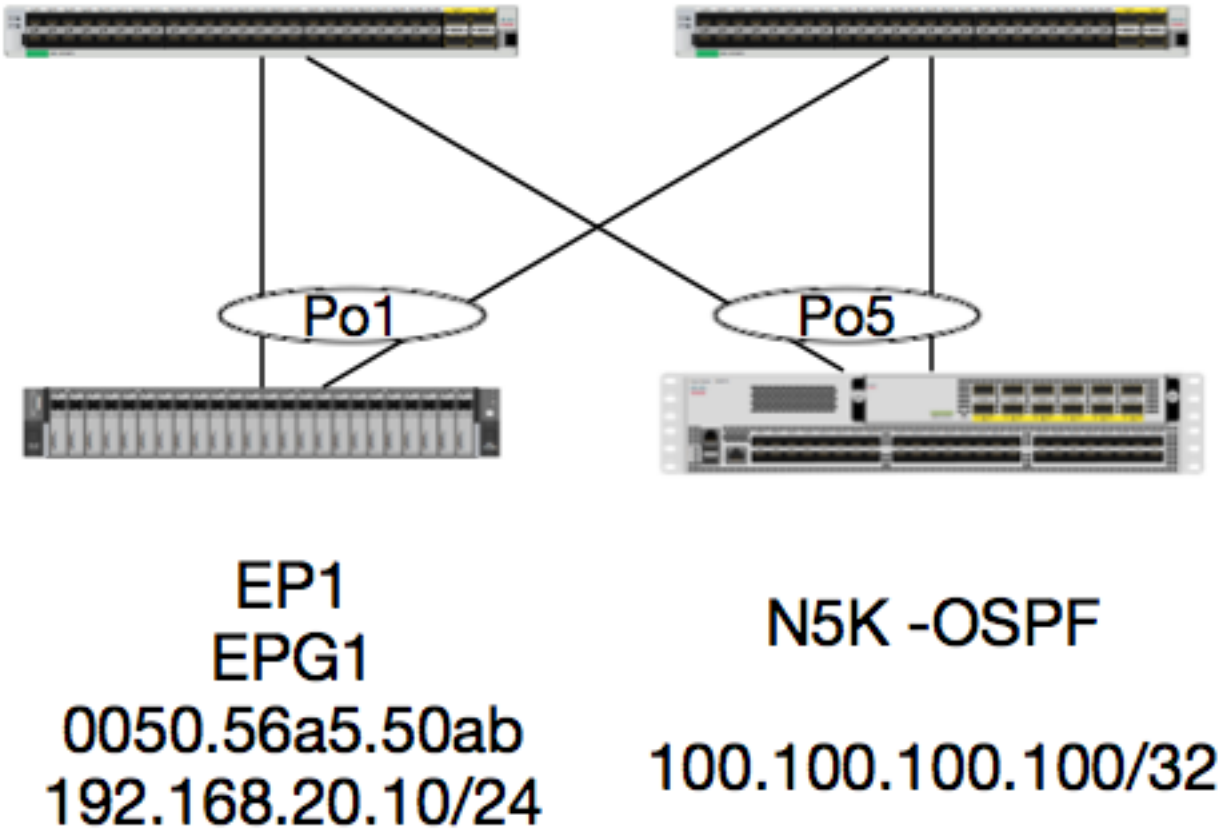
```
module-1(DBG-TAH-elam-insel9)# show platform internal hal tunnel rtep apd Non-Sandbox Mode
LEGEND: ------- ifId: Interface Id IP: IP address HwVrfId: Hardware Vrf Id SrcTepIdx: Source Tep
Index BDXlate: Egress BDXlate DstInfoIdx: Destination info index RwEncapIdx: Rw Encap Index
```

```
ECMPIdx: ECMP Index Num: Number of hops ECMPMbrIdx: ECMP member Index L2 Index: L2 Index
RwDmacIdx: Rw Dmax Index Num. of Sandboxes: 1 Sandbox_ID: 0, BMP: 0x0 Remote Tep Count: 15
================================================================================================
============================== ifId IP HwVrfId BDXlate SrcTepIdx DstInfoIdx RwEncapIdx ECMPIdx
ECMPMbrIdx Num L2Index RwDmacIdx
================================================================================================
============================== 18010002 192.168.120.67 2 1 3a9a 3005 6 0 0 2 1a030000 0 <----
RwEncapIdx is 6! Same as the "encap_idx" in the ELAM Report. 1a031000 1
```

此通道有RwEncapIdx (重写Encap索引) 6，是什么在elam显示。

1个EP --> L3外的已路由流

拓扑



在本例中，我们将跟踪一数据包的数据包流从发送ICMP的EP1的对在运行OSPF的N5K的一环回。 N5K通过在同一个对的L3Out连接EX交换机。

因为我们验证编程在本文初的本地EP，假设EP在硬件方面正确地学习和继续到路由验证。

首先，请检查OSPF状态和路由表：

```
leaf6# show ip ospf neighbors vrf jr:sb OSPF Process ID default VRF jr:sb Total number of
neighbors: 2 Neighbor ID Pri State Up Time Address Interface 27.27.27.1 1 FULL/BDR 00:22:39
10.10.27.1 Vlan28 <---- Leaf5 27.27.27.3 1 FULL/DROTHER 00:22:37 10.10.27.3 Vlan28 <---- N5K
leaf6# show ip route vrf jr:sb 100.100.100.100 IP Route Table for VRF "jr:sb" '*' denotes best
ucast next-hop '**' denotes best mcast next-hop '[x/y]' denotes [preference/metric] '%<string>'
in via output denotes VRF <string> 100.100.100.100/32, ubest/mbest: 1/0 *via 10.10.27.3, vlan28,
[110/5], 00:16:58, ospf-default, intra
```

因此我们知道routng表表示下一跳作为5K在10.10.27.3。 好开始，但是如何能验证什么硬件有？

在硬件方面请首先检查邻接表确保我们有ARP被解决对10.10.27.3，并且那编程与正确接口：

```
leaf6# vsh_lc module-1# show forwarding adjacency IPv4 adjacency information, adjacency count 20
next-hop rewrite info interface phy i/f ------------- ------------- -------------- ----------
----- 10.10.27.1 0022.bdf8.19ff Vlan28 Tunnel3 10.10.27.3 8c60.4f02.88fc Vlan28 port-channel5
```

MAC地址匹配在5K ：

```
ACI-5548-B# show interface vlan 3117 Vlan3117 is up, line protocol is up Hardware is EtherSVI,
address is 8c60.4f02.88fc Internet Address is 10.10.27.3/29 MTU 1500 bytes, BW 1000000 Kbit, DLY
10 usec
```

在EX平台上，有分配到VRF的“hw_vrf_idx”。 当我们验证硬件编程，此索引将被参考。 请查找索引：

```
module-1# show system internal eltmc info vrf jr:sb VRF-TABLE: jr:sb vrf_type: tenant :::
context_id: 6 overlay_index: 0 ::: vnid: 2129921 scope: 5 ::: sclass: 16386 v4_table_id: 0x5 :::
v6_table_id: 0x80000005 intf_count: 5 ::: intrn_vlan_id: 0 VRF Intf: Vlan11 ::: src_plcy_incomp:
0 vnid_hex: 0x208001 ::: ingress_policy: 0x1 vrf_intf_list:
Vlan28,Vlan16,Vlan9,Vlan11,loopback2, hw_vrf_idx: 4612 ::: nb_egr_outer_bd: 0 sb_egr_outer_bd: 0
vrf_bd_list: 28,16,11,9, sb_egr_outer_bd: 0 ::: sdk_vrf_id: 5 [SDK Info]: vrf_name: jr:sb
vrf_id: 5 ::: hw_vrf_idx: 4612 vrf_vnid: 2129921 ::: is_infra: 0 tornbinfrahwbd: 0 :::
torsbinfrahwbd: 0 ingressBdAclLabel: 0 ::: ingBdAclLblMask: 0 egressBdAclLabel: 0 :::
egrBdAclLblMask: 0 sg_label: 5 ::: sclass: 16386 sp_incomplete: 1 ::: sclassprio: 3 [SDB INFO]:
v4 table vrf type: 1 vrf id: 5 vnid: 2129921 internal infra vlan: 0 external router
mac:00:22:bd:f8:19:ff v6 table vrf type: 1 vrf id: 5 vnid: 2129921 internal infra vlan: 0
external router mac:00:22:bd:f8:19:ff ::::
```

在我们检测邻接后， HAL应该编程路由。 使用以下命令，我们能检查此：

```
module-1# show platform internal hal l3 routes | head ----------------------------------------
--------------------------------------------------------------------------------------------
------------------------------------------------- LEGEND: | ------------------------------
--------------------------------------------------------------------------------------------
------------------------------------------------------------------ LID: Logical ID RID: Route ID
PID: Physical ID NB-ID:Next-Base ID HIT IDX: Next-Hop HitIndex CLP : Class Priority TBI: Trie
Base Index | SC : Sup-Copy SSR: Src Sup-Redirect DSR: Dst Sup-Redirect TDD :TTL Disable NB:
NextBaseType SDC : Src Direct Connect TRO: Trie Offset | SPI: Src Policy Inc DPI: Dst Policy Inc
DR : Default Route LE :Learn Enable [E:Ecmp/A:Adj] ILL : Is Link Local ISS: Is Shared Services |
RT : Route Type FWD: Forwarding HR : Host Routes EP :Ext Prefixes DLR: Default Lpm Route CLSS:
Class Id RDEL: Route in Deletion | BNE: Bind Notify Enable SNE: Sclass Notify Enable BE : Bounce
Enable IDL :Ivxlan DoNotLearn DL : Dest Local SA : Src Only AI : Age Interval | SF : Static Flag
SH : Src Hit DH: Dest Hit | module-1# show platform internal hal l3 routes ----------------------
--------------------------------------------------------------------------------------------
------------------------------------------------------------------ LEGEND: | ----------
--------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------ LID:
Logical ID RID: Route ID PID: Physical ID NB-ID:Next-Base ID HIT IDX: Next-Hop HitIndex CLP :
Class Priority TBI: Trie Base Index | SC : Sup-Copy SSR: Src Sup-Redirect DSR: Dst Sup-Redirect
TDD :TTL Disable NB: NextBaseType SDC : Src Direct Connect TRO: Trie Offset | SPI: Src Policy
Inc DPI: Dst Policy Inc DR : Default Route LE :Learn Enable [E:Ecmp/A:Adj] ILL : Is Link Local
ISS: Is Shared Services | RT : Route Type FWD: Forwarding HR : Host Routes EP :Ext Prefixes DLR:
Default Lpm Route CLSS: Class Id RDEL: Route in Deletion | BNE: Bind Notify Enable SNE: Sclass
Notify Enable BE : Bounce Enable IDL :Ivxlan DoNotLearn DL : Dest Local SA : Src Only AI : Age
Interval | SF : Static Flag SH : Src Hit DH: Dest Hit | ----------------------------------------
--------------------------------------------------------------------------------------------
------------------------------------------------------ | | | | | LID |<-----------------
Trie ------------>|<Dleft Trie>| | VRF | Prefix/Len | RT| RID | LID | Type| PID | FPID/| HIT
|N| NB-ID | NB Hw | PID | FPID/| TBI |TRO|Ifindex|CLSS|CLP| AI |SH|DH| Flags | |-----|----------
---------------------------------|---|-----|----------|-----| | TID | IDX |B| | Idx | | TID |---
------|---|-------|----|---|----|--|--|-----------------| |-----|----------------------------
-------------|---|-----|----------|-----|<----------------- DLEFT ------------>|-----|------|--
-------|---|-------|----|---|----|--|--|-----------------| | | | | | | PID | FPID/| HIT |N|
NB-ID | NB Hw | | | | | | | | | | | | | | | | | | TID | IDX |B| | Idx | | | | | | | | | |
```

```
| | | | | | | |<------------------ TCAM ------------>| | | | | | | | | | | | | | | | | PID
| TCAM | HIT |N| NB-ID | NB Hw | | | | | | | | | | | | | | | | | | ID | IDX |B| | Idx | | |
| | | | | | | | | ---------------------------------------------------------------------------
---------------------------------------------------------------------------------------------
----------------- |Sandbox_ID: 0 Asic Bitmap: 0x0 ------------------------------------------
---------------------------------------------------------------------------------------------
------------------------------------------- module-1# show platform internal hal l3 routes |
egrep 100.100.100.100 | 4612| 100.100.100.100/ 32| UC| e4| 4a04| TRIE| 10| 5/ 0| 6010|A| 7567|
802e| 186a| 1/ 2| 10| 0| 0| f| 3| 0| 0| 0|spi,dpi
```

此输出提供我们关于下一跳路由的信息。 4612是小的hw_vrf_idx ：sb VRF。 为了我们验证下一跳，“NB在TCAM的Hw Idx”将使用下个表：

```
module-1# show platform internal hal l3 nexthops Non-Sandbox Mode LEGEND: ------- NHOP ID : Nhop
Identifier (Hex) CONS : H/W S/W info Consistency TYPE : Nexthop Type ACTN : Nexthop Action Vrf :
L3 Vrf of the Nhop L3 INTF : L3 interface index (Hex) L2 INTF : L2 interface index (Hex) BDID Or
RwVRF : Bridge Domain Id Or Rewrite Vrfid (Hex) INFR : ACI Infra valid PVRF : Preserve VRF LRN :
Learn Enabled VRFR : VRF Rewrite PID : Physical ID FPID : FP of this nexthop TLID : Tile Id
within FP HIT IDX : Location of this Nhop (Hex) Mac Entry: TYP : Type INTF : Interface related
Info (Hex) LRN : Learn Info DL : Destination Local MLD : Unused VNB : Vnid use BD DFL : Default
Entry VLD : MacKey Valid FT : FID Type FV : FID Valid FID : FID value (Hex) Mac : L2 MAC Address
L2 Ifabric Info: CLSS : Source Class CLP : Source Class Priority EPG : EndPoint Group BNE : Bind
Notification Enabled SNE : Source Address Notification Enabled CNE : Source class Notification
Enabled DL : iVxlan DL SPI : Source Policy Incomplete DPI : Dest Policy Incomplete IP Address :
IP address Sandbox_ID: 0 Asic Bitmap: 0x0 Summary info for 31 L3 Nexthop objects C T A BDID I P
V T |---------------------Mac Entry-------------------|----L2 Ifabric Info----| NHOP O Y C L3 L2
Or N V L R L HIT|T L M V D V|--------------Mac Key-------| C B S C S D| ID N P T INTF INTF RwVRF
F R R F FP I IDX|Y INTF R D L N F L|F F FID | L N N N D P P| (Hex) S E N Vrf (H) (H) (H) R F N R
PID ID D (H)|P (H) N L D B L D|T V (H) Mac |CLSS P EPG E E E L I I| IP Address --------+-+-+-+--
--+--------+--------+----+-+-+-+-+----+--+-+----+-+------+-+-+-+-+-+-+-+-+------+--------------
--+----+-+----+-+-+-+-+-+-+---------------------------- module-1# show platform internal hal l3
nexthops | grep 802e 7567 N I F 5 901001c 16000004 1c 0 0 0 0 2e 9 0 802e 0 22 0 0 0 0 0 1 1 1
1214 8c:60:4f:02:88:fc 0 0 2c0d 0 0 0 0 0 0 10.10.27.3
```

这里，我们采取“NB Hw Idx”并且映射它对“HIT IDX”。 这显示我们条目与下一跳MAC/IP相应。 这是查看"l3 defip等同显示”并且"l3出口在生成1 ACI分支交换机的Broadcom显示”。

我们能看到，表有正确信息：

L2 INTF ：0x16000004 ---> IfIndex Port-Channel 5

HIT IDX ： 从Nb驱动的索引在hal l3路由的Hw Idx

MAC ： 8c:60:4f:02:88:fc -->下一跳SVI MAC在5K的

EPG ： L3 EPG SCLASS

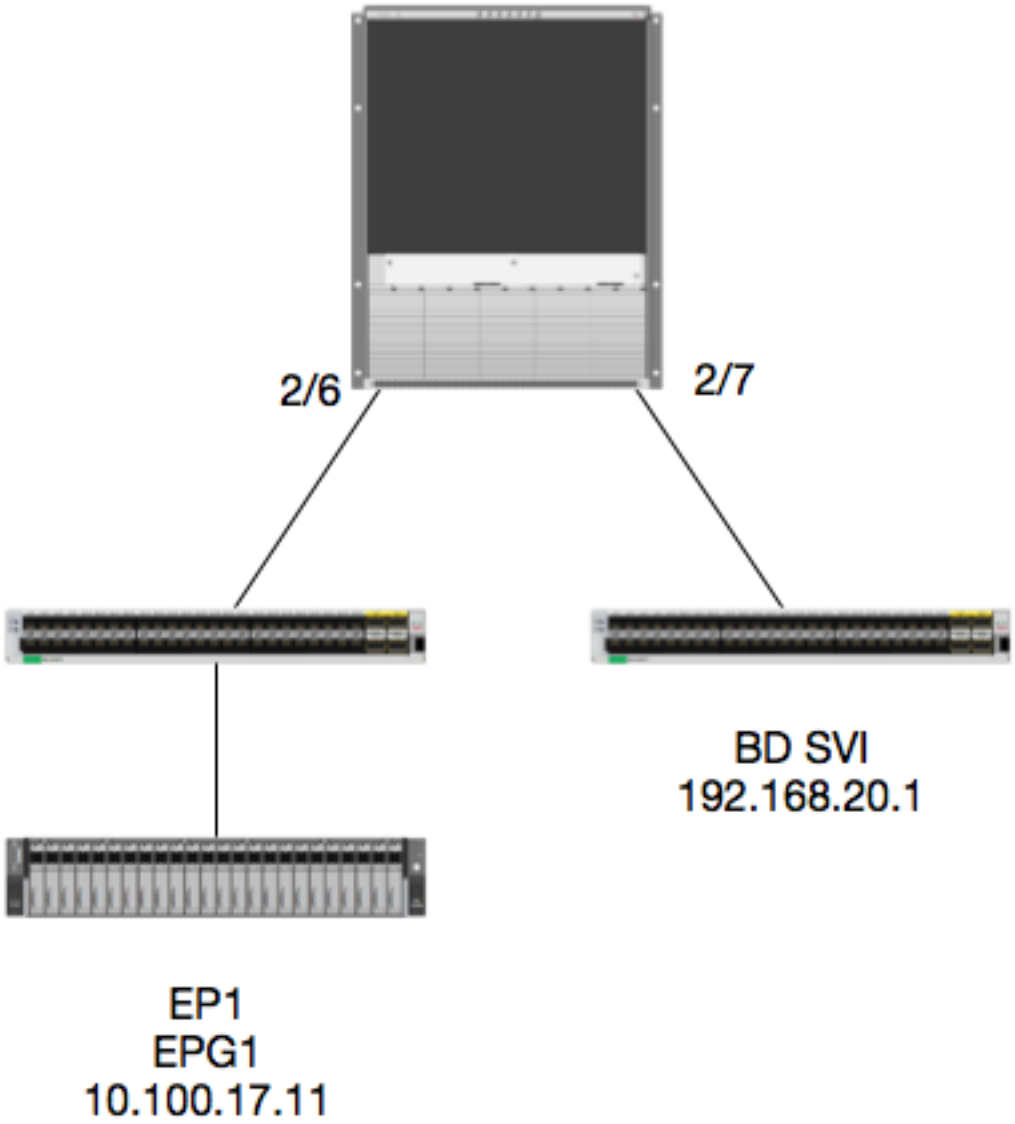
IP 地址：10.10.27.3 ---> SVI下一跳IP在5K的

伊拉姆

```
leaf6# pwd
/var/sysmgr/tmp_logs

leaf6# cat elam_report.txt | grep ip.da
    sug_pr_lu_vec_l3v.ip.da: 0x0000000000000000064646464 leaf6# cat elam_report.txt | grep ip.sa
sug_pr_lu_vec_l3v.ip.sa: 0x00000000000000000C0A8140A leaf6# cat elam_report.txt | grep adj
sug_lurw_vec.dst_addr.adj: 0x8C604F0288FC sug_lurw_vec.dst_addr.adj.padfield: 0x04F0288FC
sug_lurw_vec.dst_addr.adj.idx: 0x2318 sug_lurw_vec.adj_vld: 0x0 leaf6# cat elam_report.txt |
grep macdarslt.hit_idx sug_fpc_lookup_vec.fplu_vec.rslt.macdarslt.hit_idx: 0x802E
```

1个EP -->远程EP或SVI -脊椎验证

拓扑



逻辑

在本例中，我们将跟踪一数据包的数据包流从EP1的被注定对远程BD交换的虚拟接口(SVI)。 此示例目的将将验证脊椎转发保证数据包发送对正确分支。 假设数据包发送给入口分支的脊椎代理。

在脊椎，请首先验证委员会Oracles协议(小屋)目的地IP的，因为数据包发送给查找的脊椎代理：

```
calo1-spine1# show coop internal info ip-db | grep -A 10 192.168.20.1 <------- IP address :
192.168.20.1 Vrf : 2129921 Flags : 0 EP vrf vnid : 2129921 EP IP : 192.168.20.1 Publisher Id :
10.0.224.88 Record timestamp : 11 04 2016 16:41:16 422062712 Publish timestamp : 11 04 2016
16:41:16 424633605 Seq No: 0 Remote publish timestamp: 01 01 1970 00:00:00 0 URIB Tunnel Info
Num tunnels : 1 Tunnel address : 10.0.224.88 <---- REMOTE LEAF Tunnel ref count : 1
```

请验证什么分支有该TEP地址：

```
spine1# acidiag fnvread | grep 10.0.224.88 105 1 calo1-leaf5 FDO20160TPS 10.0.224.88/32 leaf
active 0
```

因为我们知道数据包进入在模块2的脊椎，波尔特6，我们能附加到模块2并且查看波尔特布局。

```
spine1# vsh Cisco iNX-OS Debug Shell This shell should only be used for internal commands and
exists for legacy reasons. User should use ibash infrastructure as this will be deprecated.
calo1-spine1# attach module 2 Attaching to module 2 ... To exit type 'exit', to abort type '$.'
No directory, logging in with HOME=/ Bad terminal type: "xterm-256color". Will assume vt100.
Cisco iNX-OS Debug Shell This shell should only be used for internal commands and exists for
legacy reasons. User should use ibash infrastructure as this will be deprecated. Loading parse
tree (LC). Please be patient... module-2# module-2# show platform internal hal l2 port gpd
Legend: ------- IfId: Interface Id IfName: Interface Name I P: Is PC Mbr IfId: Interface Id Uc
PC Cfg: UcPcCfg Idx Uc PC MbrId: Uc Pc Mbr Id As: Asic AP: Asic Port Sl: Slice Sp: Slice Port
Ss: Slice SrcId Ovec: Ovector (slice | srcid) L S: Local Slot Reprogram: L3: Is L3 P: PifTable
Xla Idx: Xlate Idx RP: Rw PifTable Ovx Idx: OXlate Idx IP: If Profile Table N L3: Num. of L3 Ifs
RS: Rw SrcId Table NI L3: Num. of Infra L3 Ifs DP: DPort Table Vif Tid: Vif Tid SP: SrcPortState
Table RwV Tid: RwVif Tid RSP: RwSrcPortstate Table Ing Lbl: Ingress Acl Label UC: UCPcCfg Egr
Lbl: Egress Acl Label UM: UCPcMbr Reprogram: PROF ID: Lport Profile Id VS: VifStateTable HI:
LportProfile Hw Install RV: Rw VifTable Num. of Sandboxes: 1 Sandbox_ID: 0, BMP: 0x0 Port Count:
7
================================================================================================
========================================== Uc Uc | Reprogram | | Rep | I PC Pc L | R I R D R U
U X | L Xla Ovx N NI Vif RwV Ing Egr | V R | PROF H IfId Ifname P Cfg MbrID As AP Sl Sp Ss Ovec
S | P P P S P Sp Sp C M L | 3 Idx Idx L3 L3 Tid Tid Lbl Lbl | S V | ID I
================================================================================================
========================================== 1f5 SpInBndMgmt 0 9de 1a 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 D-2d4 D-3e1 0 0 0 0 1 0 1a080000 Eth2/1 0 9a 1c 0 11 0 10 20 20 1 0 0 0 0 0 0
0 0 0 0 1 b b 1 1 D-f3 D-61 100 0 0 0 1 0 1a081000 Eth2/2 0 9b 22 0 d 0 c 18 18 1 0 0 0 0 0 0 0
0 0 0 1 c c 1 1 D-1ee D-30b 100 0 0 0 1 0 1a084000 Eth2/5 0 9e 1e 0 3d 1 14 28 a8 1 0 0 0 0 0 0
0 0 0 0 1 1 1 1 1 D-19a D-2ee 100 0 0 0 1 0 1a085000 Eth2/6 0 9f 24 0 39 1 10 20 a0 1 0 0 0 0 0
0 0 0 0 0 1 e e 1 1 D-87 D-184 100 0 0 0 1 0 <--- Interface that connects to Leaf 6 is on ASIC 0
SLICE 1 1a086000 Eth2/7 0 a0 26 0 35 1 c 18 98 1 0 0 0 0 0 0 0 0 0 1 d d 1 1 D-1d0 D-357 100 0
0 0 1 0 1a088000 Eth2/9 0 a2 20 1 d 0 c 18 18 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 D-3ea D-1a9 100 0
0 0 1 0
```

现在我们知道运行我们的伊拉姆的哪个ASIC的。 ASIC 0。

```
module-2# debug platform internal tah elam asic 0 module-2(DBG-TAH-elam)# trigger reset module-
2(DBG-TAH-elam)# trigger init in-select 13 out-select 0 module-2(DBG-TAH-elam-insel13)# set
inner ipv4 src_ip 10.100.17.11 dst_ip 192.168.20.1 module-2(DBG-TAH-elam-insel13)# start stat
module-2(DBG-TAH-elam-insel13)# stat ELAM STATUS ========== Asic 0 Slice 0 Status Armed Asic 0
Slice 1 Status Armed module-2(DBG-TAH-elam-insel13)# stat ELAM STATUS ========== Asic 0 Slice 0
Status Triggered <---- Packet triggered from FM Asic 0 Slice 1 Status Triggered <---- Packet
triggered from Front Panel
```

查看伊拉姆，我们能找到ovector索引：

```
Front Panel ELAM drove sug_elam_out_sidebnd_no_spare_vec.ovector_idx: 0xB8
```

现在，如何映射0xb8对端口？ 因为我们知道数据包应该获得发送对一个结构模块(FM)查找的，我们能查看端口映射查找目的FM ：

```
module-2# show platform internal hal l2 internal-port pi Num. of Sandboxes: 1 Legend: -------
IfId: Interface Id IfName: Interface Name As: Asic AP: Asic Port Sl: Slice SP: Slice Port Ss:
Slice SrcId Ovec: Ovector UcPcCfgId: Uc Pc CfgId Lb Mbrid: LB MbrId Sandbox_ID: 0, BMP: 0x0
Internal Port Count: 32 ========================================================================
UcPc Lb IfId IfName As AP Sl SP Ss Ovec CfgId MbrId
======================================================================== 7d - 0 21 0 20 38 38 0 4
7e - 0 29 1 0 0 80 0 8 7f - 1 21 0 20 38 38 0 c 80 - 1 29 1 0 0 80 0 10 81 - 2 21 0 20 38 38 0
14 82 - 2 29 1 0 0 80 0 18 83 - 3 21 0 20 38 38 0 1c 84 - 3 29 1 0 0 80 0 20 95 - 0 19 0 18 30
30 0 3 96 - 0 49 1 20 38 b8 0 7 <------- Using ASIC0 / Ovec B8, we get MbrId 7, Slice does not
matter 97 - 1 19 0 18 30 30 0 b 98 - 1 49 1 20 38 b8 0 f 99 - 2 19 0 18 30 30 0 13 9a - 2 49 1
20 38 b8 0 17 9b - 3 19 0 18 30 30 0 1b 9c - 3 49 1 20 38 b8 0 1f ad - 0 25 0 24 40 40 0 1 ae -
0 41 1 18 30 b0 0 6 af - 1 25 0 24 40 40 0 9 b0 - 1 41 1 18 30 b0 0 e b1 - 2 25 0 24 40 40 0 11
b2 - 2 41 1 18 30 b0 0 16 b3 - 3 25 0 24 40 40 0 19 b4 - 3 41 1 18 30 b0 0 1e dd - 0 15 0 14 28
28 0 2 de - 0 4d 1 24 40 c0 0 5 df - 1 15 0 14 28 28 0 a e0 - 1 4d 1 24 40 c0 0 d e1 - 2 15 0 14
28 28 0 12 e2 - 2 4d 1 24 40 c0 0 15 e3 - 3 15 0 14 28 28 0 1a e4 - 3 4d 1 24 40 c0 0 1d
```

此MbrId是在USD的接口该地图对在FM的一个接口。 我们能发现FM通过查看USD建立接口和检查

波尔特7 ：

```
module-2# show platform internal usd port info | grep -A 3 "Int 7" Port 73.0 (Int 7) : Admin UP
Link UP Remote slot22.asic0 slice:1 slice port:32 lcl srcid:56 gbl srcid:184 asic mrl:0xd07c010,
mac mrl:0x12c84010, mac:16, chan:0 speed 106G serdes: 0x328 0x329 0x32a 0x32b
```

“slot”是基于的0，并且FM编号是基于的1，因此我们需要添加1到列出的编号此处。 这意味着应该发送数据包到FM 23。

集成IP

正如在高山，有作为外面IP地址用于的集成IP确定小屋查找的哈希。 为了查找此，您需要运行此命令和grep内在DST IP:的

```
module-2(DBG-TAH-elam-insel7)# show forwarding route synthetic vrf all | grep 192.168.20.1
SYNTH-88 1.203.211.185/32 0x208001 192.168.20.1
```

这表示我们， 1.203.211.185是我们的集成IP。 基于此，我们能也设置“外面DST IP”在我们的FM elam是这。 我们在FM应该触发：

结构模块伊拉姆

```
module-23(DBG-TAH-elam-insel7)# trigger reset module-23(DBG-TAH-elam)# trigger init in-select 13
out-select 0 module-23(DBG-TAH-elam-insel13)# set outer ipv4 dst_ip 1.203.211.185 <----- DST IP
IS THE SYNTHETIC IP module-23(DBG-TAH-elam-insel13)# set inner ipv4 src_ip 10.100.17.11 dst_ip
192.168.20.1 module-23(DBG-TAH-elam-insel13)# start stat module-23(DBG-TAH-elam-insel13)# stat
ELAM STATUS ========== Asic 0 Slice 0 Status Armed Asic 0 Slice 1 Status Armed Asic 0 Slice 2
Status Armed Asic 0 Slice 3 Status Armed Asic 0 Slice 4 Status Armed Asic 0 Slice 5 Status Armed
module-23(DBG-TAH-elam-insel13)# stat ELAM STATUS ========== Asic 0 Slice 0 Status Armed Asic 0
Slice 1 Status Armed Asic 0 Slice 2 Status Triggered <---- Triggered on SLICE 2 Asic 0 Slice 3
Status Armed Asic 0 Slice 4 Status Armed Asic 0 Slice 5 Status Armed
```

明显地，请转存报告全文，但是请查看ovector_idx为我们触发的此数据包：

lac_elam_out_sidebnd_no_spare_vec.ovector_idx ：0x20 **<-----**用于下面命令的Ovector索引

我们如何推测接口有该ovector ？。 在FM，请运行此：

```
module-23(DBG-TAH-elam-insel13)# show platform internal hal l2 port gpd Legend: ------- IfId:
Interface Id IfName: Interface Name I P: Is PC Mbr IfId: Interface Id Uc PC Cfg: UcPcCfg Idx Uc
PC MbrId: Uc Pc Mbr Id As: Asic AP: Asic Port Sl: Slice Sp: Slice Port Ss: Slice SrcId Ovec:
Ovector (slice | srcid) L S: Local Slot Reprogram: L3: Is L3 P: PifTable Xla Idx: Xlate Idx RP:
Rw PifTable Ovx Idx: OXlate Idx IP: If Profile Table N L3: Num. of L3 Ifs RS: Rw SrcId Table NI
L3: Num. of Infra L3 Ifs DP: DPort Table Vif Tid: Vif Tid SP: SrcPortState Table RwV Tid: RwVif
Tid RSP: RwSrcPortstate Table Ing Lbl: Ingress Acl Label UC: UCPcCfg Egr Lbl: Egress Acl Label
UM: UCPcMbr Reprogram: PROF ID: Lport Profile Id VS: VifStateTable HI: LportProfile Hw Install
RV: Rw VifTable Num. of Sandboxes: 1 Sandbox_ID: 1, BMP: 0x1 Port Count: 8
================================================================================================
========================================== Uc Uc | Reprogram | | Rep | I PC Pc L | R I R D R U
U X | L Xla Ovx N NI Vif RwV Ing Egr | V R | PROF H IfId Ifname P Cfg MbrID As AP Sl Sp Ss Ovec
S | P P P S P Sp Sp C M L | 3 Idx Idx L3 L3 Tid Tid Lbl Lbl | S V | ID I
================================================================================================
========================================== ae fc0-lc1:0-0 1 0 3 0 11 0 10 20 20 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 - - 0 0 0 0 0 0 <----------- Interface points to LC1 ASIC 0 / SLICE 0 af fc0-
lc1:0-1 1 0 4 0 3d 2 c 18 98 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - - 0 0 0 0 0 0 b0 fc0-lc1:1-0 1 0
13 0 d 0 c 18 18 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - - 0 0 0 0 0 0 b1 fc0-lc1:1-1 1 0 14 0 39 2 8
10 90 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - - 0 0 0 0 0 0 b2 fc0-lc1:2-0 1 0 23 0 5d 3 14 28 e8 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - - 0 0 0 0 0 0 b3 fc0-lc1:2-1 1 0 24 0 21 1 8 10 50 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 - - 0 0 0 0 0 0 b4 fc0-lc1:3-0 1 0 33 0 51 3 8 10 d0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 - - 0 0 0 0 0 0
```

该ovector映射对LC1 (在slot 2的线卡，因为是基于的0)，在ASIC 0/片式0。 我们从在LC最初运行的伊拉姆知道，我们在此片式触发了：

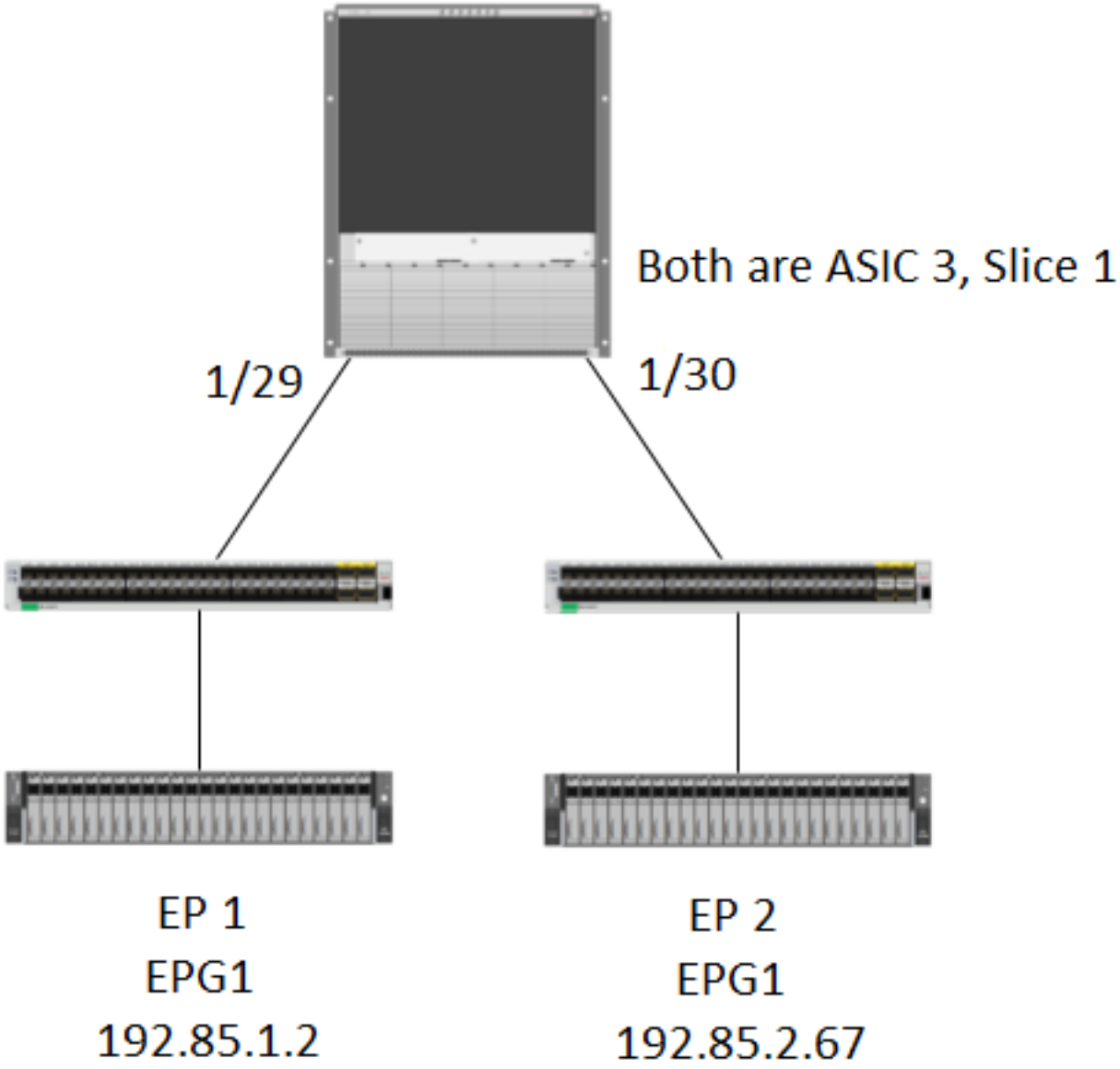
```
module-2# debug platform internal tah elam asic 0
module-2(DBG-TAH-elam)# trigger reset
module-2(DBG-TAH-elam)# trigger init in-select 13 out-select 0
module-2(DBG-TAH-elam-insel13)# set inner ipv4 src_ip 10.100.17.11 dst_ip 192.168.20.1
module-2(DBG-TAH-elam-insel13)# start
stat
module-2(DBG-TAH-elam-insel13)# stat
ELAM STATUS
===========
Asic 0 Slice 0 Status Armed
Asic 0 Slice 1 Status Armed

module-2(DBG-TAH-elam-insel13)# stat
ELAM STATUS
===========
Asic 0 Slice 0 Status Triggered <---- Packet triggered from FM Asic 0 Slice 1 Status Triggered
<---- Packet triggered from Front Panel
```

在此伊拉姆的ovector是sug_elam_out_sidebnd_no_spare_vec.ovector_idx ：0x98，我们从“hal l2端口gpd”认识，映射对在LC的正确接口：

```
================================================================================================================
=========================================
                       Uc   Uc                          |       Reprogram       |
| Rep |
                    I PC   Pc                       L |   R I R D    R  U U X | L Xla Ovx N
NI Vif    RwV    Ing  Egr  | V R | PROF H
IfId     Ifname      P Cfg  MbrID As AP Sl Sp Ss Ovec S | P P P S P Sp Sp C M L | 3 Idx Idx L3
L3 Tid    Tid    Lbl  Lbl  | S V | ID   I
================================================================================================================
=========================================
1f5      SpInBndMgmt 0 9de  1a    0  0  0  0  0  0    0   0 0 0 0 0 0  0  0 0 0   0 0   0   0  0
D-2d4  D-3e1  0    0      0 0   1    0
1a080000 Eth2/1      0 9a   1c    0  11 0  10 20 20   1   0 0 0 0 0 0  0  0 0 0   1 b   b   1  1
D-f3   D-61   100  0      0 0   1    0
1a081000 Eth2/2      0 9b   22    0  d  0  c  18 18   1   0 0 0 0 0 0  0  0 0 0   1 c   c   1  1
D-1ee  D-30b  100  0      0 0   1    0
1a084000 Eth2/5      0 9e   1e    0  3d 1  14 28 a8   1   0 0 0 0 0 0  0  0 0 0   1 1   1   1  1
D-19a  D-2ee  100  0      0 0   1    0
1a085000 Eth2/6      0 9f   24    0  39 1  10 20 a0   1   0 0 0 0 0 0  0  0 0 0   1 e   e   1  1
D-87   D-184  100  0      0 0   1    0
1a086000 Eth2/7 0 a0 26 0 35 1 c 18 98 1 0 0 0 0 0 0 0 0 0 0 1 d d 1 1 D-1d0 D-357 100 0 0 0 1 0
<--- Interface that connects to Leaf 5 1a088000 Eth2/9 0 a2 20 1 d 0 c 18 18 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 D-3ea D-1a9 100 0 0 0 1 0
```

额外的方案：获得不在“hal内部端口pi”输出中的Ovector

拓扑



逻辑

有我们捉住一数据包不安排在“**显示平台内部hal l2内部端口pi的**” Ovector表的一些方案。 在下面的方案中，我们实际上捉住回来从FM的数据包，因此我们需要查看一个不同的表发现哪个前面板端口数据包选择。

注意以上的拓扑是了解中转流量的一个完全不同的环境(没有代理路由)。 模块是N9K-X9732C-EX。

```
@module-1# debug platform internal tah elam asic 3
@module-1(DBG-elam)# trigger reset
@module-1(DBG-elam)# trigg init in-select 13 out-select 0
@module-1(DBG-elam-insel13)# set inner ipv4 src_ip 192.85.1.2 dst_ip 192.85.2.67
@module-1(DBG-elam-insel13)# star
@module-1(DBG-elam-insel13)# stat
ELAM STATUS
===========
Asic 3 Slice 0 Status Armed
Asic 3 Slice 1 Status Triggered

@module-1(DBG-elam-insel13)# report | grep ovector
```

```
 sug_elam_out_sidebnd_no_spare_vec.ovector_idx: 0xA0 <<<<<<<<<<<<<<<< now we look for this in
the "hal internal-port pi" command


@module-1# show platform internal hal l2 internal-port pi
No sandboxes exist
Num. of Sandboxes: 1
Legend:
-------
IfId:      Interface Id               IfName:     Interface Name
As:        Asic                       AP:         Asic Port
Sl:        Slice                      SP:         Slice Port
Ss:        Slice SrcId                Ovec:       Ovector
UcPcCfgId: Uc Pc CfgId                Lb Mbrid:   LB MbrId

Sandbox_ID: 0, BMP: 0x0
 Internal Port Count: 24

=====================================================================
                                                      UcPc  Lb
IfId     IfName                        As AP Sl SP Ss Ovec CfgId MbrId
=====================================================================
7d       -                             0  21 0  20 38 38   0     4
7e       -                             0  29 1  0  0  80   0     8
7f       -                             1  21 0  20 38 38   0     c
80       -                             1  29 1  0  0  80   0     10
81       -                             2  21 0  20 38 38   0     14
82       -                             2  29 1  0  0  80   0     18
83       -                             3  21 0  20 38 38   0     1c
84       -                             3  29 1  0  0  80   0     20
ad       -                             0  25 0  24 40 40   0     1
ae       -                             0  41 1  18 30 b0   0     6
af       -                             1  25 0  24 40 40   0     9
b0       -                             1  41 1  18 30 b0   0     e
b1       -                             2  25 0  24 40 40   0     11
b2       -                             2  41 1  18 30 b0   0     16
b3       -                             3  25 0  24 40 40   0     19
b4       -                             3  41 1  18 30 b0   0     1e
dd       -                             0  15 0  14 28 28   0     2
de       -                             0  4d 1  24 40 c0   0     5
df       -                             1  15 0  14 28 28   0     a
e0       -                             1  4d 1  24 40 c0   0     d
e1       -                             2  15 0  14 28 28   0     12
e2       -                             2  4d 1  24 40 c0   0     15
e3       -                             3  15 0  14 28 28   0     1a
e4       -                             3  4d 1  24 40 c0   0     1d  <<<<<<<  we cant find an
entry that matches 0xA0


@module-1# show platform internal hal l2 port gpd
Legend:
-------
<snip>

Sandbox_ID: 0, BMP: 0x0
 Port Count: 6

=================================================================================================
==========================================
                      Uc   Uc                          |       Reprogram
|                                       | Rep |
                    I PC   Pc                        L |   R I R D    R  U U X | L Xla Ovx N  NI
Vif    RwV    Ing  Egr  | V R | PROF H
IfId     Ifname      P Cfg  MbrID As AP Sl Sp Ss Ovec S | P P P S P Sp Sp C M L | 3 Idx Idx L3
```

```
L3 Tid    Tid    Lbl  Lbl  | S V | ID   I
=============================================================================================================
=============================================
1f5       SpInBndMgmt 0 9de  1a    0  0  0  0  0  0    0   0 0 0 0 0 0  0  0 0 0   0 0   0   0
0  D-2d4  D-3e1  0    0      0 0   1    0
1a000000 Eth1/1      0 1b   1c    0  11 0  10 20 20   1   0 0 0 0 0 0  0  0 0 0   1 1   1   1
1  D-13b  D-33b  500  0      1 0   3    0
1a01c000 Eth1/29     0 37   1e    3  3d 1  14 28 a8   1   0 0 0 0 0 0  0  0 0 0   1 8   8   1
1  D-3f2  D-7a   100  0      0 0   2    0
1a01d000 Eth1/30     0 38   20    3  39 1  10 20 a0   1   0 0 0 0 0 0  0  0 0 0   1 5   5   1
1  D-36e  D-362  100  0      0 0   2    0  <<<<<<< 1/30 is the phys interface that connects to
leaf 102, verified by topology, ASIC 3, Slice 1
1a01e000 Eth1/31     0 39   22    3  35 1  c  18 98   1   0 0 0 0 0 0  0  0 0 0   1 9   9   1
1  D-273  D-8    100  0      0 0   2    0
1a01f000 Eth1/32     0 3a   24    3  31 1  8  10 90   1   0 0 0 0 0 0  0  0 0 0   1 a   a   1
1  D-154  D-5d   100  0      0 0   2    0
```