

Determine Packet Flow Through an ACI Fabric

TAC

Document ID: 118930

Contributed by Joseph Ristaino, Cisco TAC Engineer.
Apr 21, 2015

Contents

Introduction

Prerequisites

Requirements

Components Used

Determine ACI Fabric Packet Flow

Single BD/Single EPG with Two Endpoints on the Same Leaf

Single BD/Single EPG with Two Endpoints on Different Leafs

Single BD/Two EPGs with One Endpoint in Each EPG on the Same Leaf

Two BDs/Two EPGs with One Endpoint in Each EPG on the Same Leaf (Routed Packet)

Introduction

This document describes how to determine the packet flow through an Application Centric Infrastructure (ACI) Fabric in various situations.

Note: All of the situations that are described in this document involve an operational ACI Fabric so that the packet flow in the hardware can be traced.

Prerequisites

Requirements

There are no specific requirements for this document.

Components Used

The information in this document is based on these hardware and software versions:

- An ACI Fabric that consists of two Spine switches and two Leaf switches
- An ESXi host with two uplinks that go to each of the Leaf switches
- An Application Policy Infrastructure Controller (APIC) that is used for initial setup

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, make sure that you understand the potential impact of any command.

Determine ACI Fabric Packet Flow

This section describes the various situations in which an ACI Fabric might be used and how to determine the packet flow.

Single BD/Single EPG with Two Endpoints on the Same Leaf

This section describes how to verify the hardware programming and packet flow for two endpoints within the same Endpoint Group (EPG)/Bridge Domain (BD) on the same Leaf switch. If the Virtual Machines (VMs) run on the same host, since they are in the same EPG, the traffic is isolated to the Virtual Switch (VS) on the host, and the traffic never has to leave the host. If the VMs run on different hosts, then the information that follows applies.

The first thing that you should verify is whether the Media Access Control (MAC) address information for both the source and destination IP addresses on the Leaf switch is learned. This is the MAC and IP address information that is used in this example:

- Source MAC address: **0050.5695.17b7**
- Source IP address: **192.168.3.2**
- Destination MAC address: **0050.5695.248f**
- Destination IP address: **192.168.3.3**

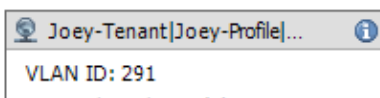
Enter the **show mac address-table** command in order to verify this information:

```
leaf2# show mac address-table
Legend:
  * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
  age - seconds since last seen, + - primary entry using vPC Peer-Link,
  (T) - True, (F) - False
VLAN      MAC Address      Type      age      Secure NTFY Ports/SWID.SSID.LIID
-----+-----+-----+-----+-----+-----+-----+-----
16        0050.5695.248f   dynamic   -        F      F      tunnel4
* 19      0050.5695.17b7   dynamic   -        F      F      eth1/31
* 19      0050.5695.248f   dynamic   -        F      F      eth1/31
```

As shown, the system learns the MAC addresses for both of the endpoints on the same VLAN. This VLAN is the Platform Independent (PI) VLAN and is locally significant to each switch. In order to verify that this is the correct PI VLAN, connect to the **vsh_lc** and enter this command into the CLI:

```
module-1# show system internal eltmc info vlan brief
VLAN-Info
VlanId  HW_VlanId  Type      Access_enc  Access_enc  Fabric_enc  Fabric_enc  BDVlan
Type
=====
9       11         BD_VLAN   Unknown     0           VXLAN      16613250    9
10      12         BD_VLAN   Unknown     0           VXLAN      15990734    10
13      13         FD_VLAN   802.1q     299        VXLAN      8507        10
16      14         BD_VLAN   Unknown     0           VXLAN      16449431    16
17      15         FD_VLAN   802.1q     285        VXLAN      8493        16
18      16         BD_VLAN   Unknown     0           VXLAN      15761386    18
19      17         FD_VLAN   802.1q     291        VXLAN      8499        18
```

The **HW_VlanId** is the VLAN that is used by the Broadcom. The **VlanId** is the PI VLAN, which maps to the **Access_enc** VLAN 291 that is derived from the VLAN pool and is the VLAN that is propagated to the Distributed Virtual Switch (DVS) Port Group:



Since this traffic flow is in the same BD and the same VLAN, the traffic should be switched locally on the Broadcom ASIC. In order to verify that the Broadcom has the correct entries in the hardware, connect to the Broadcom shell and view the Layer 2 (L2) table:

```
leaf2# bcm-shell-hw
unit is 0
Available Unit Numbers: 0
bcm-shell.0> 12 show
mac=00:22:bd:f8:19:ff vlan=19 GPORT=0x7f modid=2 port=127 Static
mac=00:50:56:95:68:c4 vlan=25 GPORT=0x5f modid=0 port=95/xe94 Hit
mac=00:22:bd:f8:19:ff vlan=16 GPORT=0x7f modid=2 port=127 Static
mac=00:22:bd:f8:19:ff vlan=29 GPORT=0x7f modid=2 port=127 Static
mac=00:22:bd:f8:19:ff vlan=32 GPORT=0x7f modid=2 port=127 Static
mac=00:22:bd:f8:19:ff vlan=26 GPORT=0x7f modid=2 port=127 Static
mac=00:50:56:95:24:8f vlan=17 GPORT=0x1f modid=0 port=31/xe30 Hit
mac=00:22:bd:f8:19:ff vlan=18 GPORT=0x7f modid=2 port=127 Static
mac=00:22:bd:f8:19:ff vlan=21 GPORT=0x7f modid=2 port=127 Static
mac=00:22:bd:f8:19:ff vlan=34 GPORT=0x7f modid=2 port=127 Static
mac=00:50:56:95:26:5e vlan=25 GPORT=0x5f modid=0 port=95/xe94 Hit
mac=00:50:56:95:c3:6f vlan=24 GPORT=0x5f modid=0 port=95/xe94 Hit
mac=00:50:56:95:5c:4d vlan=28 GPORT=0x1e modid=0 port=30/xe29 Hit
mac=00:22:bd:f8:19:ff vlan=12 GPORT=0x7f modid=2 port=127 Static Hit
mac=00:22:bd:f8:19:ff vlan=11 GPORT=0x7f modid=2 port=127 Static
mac=00:50:56:95:17:b7 vlan=17 GPORT=0x1f modid=0 port=31/xe30 Hit
mac=00:50:56:95:4e:d3 vlan=30 GPORT=0x1e modid=0 port=30/xe29 Hit
mac=00:22:bd:f8:19:ff vlan=14 GPORT=0x7f modid=2 port=127 Static
```

The output shows that the Broadcom ASIC programming is correct and that the traffic should switch locally in VLAN 17.

Single BD/Single EPG with Two Endpoints on Different Leafs

This section describes how to verify the hardware programming and packet flow for two endpoints within the same EPG/BD but on different Leaf switches.

The first thing that you should verify is whether the MAC address information for both the source and destination IP addresses on the Leaf switches is learned. This is the MAC and IP address information that is used in this example:

- Source MAC address: **0050.5695.17b7**
- Source IP address: **192.168.3.2**
- Destination MAC address: **0050.5695.bd89**
- Destination IP address: **192.168.3.11**

Enter the **show mac address-table** command into the CLI of both Leaf switches in order to verify this information:

```
leaf2# show mac address-table
Legend:
    * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
    age - seconds since last seen,+ - primary entry using vPC Peer-Link,
    (T) - True, (F) - False
VLAN    MAC Address      Type      age      Secure NTFY Ports/SWID.SSID.LID
-----+-----+-----+-----+-----+-----+-----
* 19     0050.5695.17b7   dynamic   -        F    F    eth1/31
* 19     0050.5695.248f   dynamic   -        F    F    eth1/31
```

```
leaf_1# show mac address-table
Legend:
    * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
    age - seconds since last seen,+ - primary entry using vPC Peer-Link,
    (T) - True, (F) - False
VLAN    MAC Address      Type      age      Secure NTFY Ports/SWID.SSID.LID
-----+-----+-----+-----+-----+-----+-----
27      0050.5695.248f   dynamic   -        F    F    tunnel7
```

```

27      0050.5695.17b7   dynamic   -      F      F      tunnel17
* 28    0050.5695.bd89   dynamic   -      F      F      eth1/25

```

As shown in the outputs, the source IP address is learned on the second Leaf switch (*leaf2*), while the destination IP address is learned on the first Leaf switch (*leaf_1*). Since these are on different Leaf switches, the traffic must be sent to the NorthStar ASIC on the second Leaf switch so that it can be sent upstream to the Spine switches. In order to follow the NorthStar logic, connect to the linecard *vsh*.

Enter this command in order to view a list of local entries:

```

leaf2# vsh_lc
module-1# show platform internal ns forwarding lst-12
error opening file
: No such file or directory

```

```

=====
TABLE INSTANCE : 0
=====

```

```

Legend:
POS: Entry Position          O: Overlay Instance
V: Valid Bit                 MD/PT: Mod/Port
PT: Pointer Type(A=Adj, E=ECMP, D=DstEncap N=Invalid)
PTR: ECMP/Adj/DstEncap/MET pointer
ML: MET Last
ST: Static                   PTH: Num Paths
BN: Bounce                   CP: Copy To CPU
PA: Policy Applied           PI: Policy Incomplete
DL: Dst Local                SP: Spine Proxy

```

```

-----
MO      SRC  P      M S      B C P P D S
POS    O VNID  Address      V DE MD/PT CLSS T PTR  L T PTH N P A I L P
-----
111 0 fd7f82 00:22:bd:f8:19:ff 1 0 00/00 1 A 0 0 1 1 0 0 0 1 0 0
131 0 f1ffde 00:22:bd:f8:19:ff 1 0 00/00 1 A 0 0 1 1 0 0 0 1 0 0
169 0 f37fd3 00:50:56:95:26:5e 1 0 00/24 4002 A 0 0 0 1 0 0 0 1 0 0
331 0 f37fd2 00:50:56:95:5c:4d 1 0 00/2e 8003 A 0 0 0 1 0 0 0 1 0 0
719 0 f3ffce 00:22:bd:f8:19:ff 1 0 00/00 1 A 0 0 1 1 0 0 0 1 0 0
945 0 f7ffae 00:22:bd:f8:19:ff 1 0 00/00 1 A 0 0 1 1 0 0 0 1 0 0
1390 0 fa7f9a 00:22:bd:f8:19:ff 1 0 00/00 1 A 0 0 1 1 0 0 0 1 0 0
1454 0 efffee 00:22:bd:f8:19:ff 1 0 00/00 1 A 0 0 1 1 0 0 0 1 0 0
1690 0 f37fd3 00:22:bd:f8:19:ff 1 0 00/00 1 A 0 0 1 1 0 0 0 1 0 0
1720 0 f37fd3 00:50:56:95:c3:6f 1 0 00/24 c002 A 0 0 0 1 0 0 0 1 0 0
1902 0 f1ffde 00:50:56:95:4e:d3 1 0 00/2e 8006 A 0 0 0 1 0 0 0 1 0 0
2176 0 f07fea 00:50:56:95:17:b7 1 0 00/0f 8004 A 0 0 0 1 0 0 0 0 0 0
2819 0 faff97 00:22:bd:f8:19:ff 1 0 00/00 1 A 0 0 1 1 0 0 0 1 0 0
3297 0 f07fea 00:22:bd:f8:19:ff 1 0 00/00 1 A 0 0 1 1 0 0 0 1 0 0

```

```

=====
TABLE INSTANCE : 1
=====

```

```

Legend:
POS: Entry Position          O: Overlay Instance
V: Valid Bit                 MD/PT: Mod/Port
PT: Pointer Type(A=Adj, E=ECMP, D=DstEncap N=Invalid)
PTR: ECMP/Adj/DstEncap/MET pointer
ML: MET Last
ST: Static                   PTH: Num Paths
BN: Bounce                   CP: Copy To CPU
PA: Policy Applied           PI: Policy Incomplete
DL: Dst Local                SP: Spine Proxy

```

```

-----
MO      SRC  P      M S      B C P P D S
POS    O VNID  Address      V DE MD/PT CLSS T PTR  L T PTH N P A I L P
-----
169 0 f37fd3 00:50:56:95:26:5e 1 0 00/24 4002 A e 0 0 1 0 0 0 0 1 0

```

```

331 0 f37fd2 00:50:56:95:5c:4d 1 0 00/2e 8003 A 9 0 0 1 0 0 0 0 1 0
1720 0 f37fd3 00:50:56:95:c3:6f 1 0 00/24 c002 A c 0 0 1 0 0 0 0 1 0
1902 0 flffde 00:50:56:95:4e:d3 1 0 00/2e 8006 A f 0 0 1 0 0 0 0 1 0
2176 0 f07fea 00:50:56:95:17:b7 1 0 00/0f 8004 A d 0 0 1 0 0 0 0 1 0
3507 0 fa7f9a 00:50:56:95:3e:ee 1 0 00/2e c005 A 10 0 0 1 0 0 0 0 1 0
3777 0 f37fd3 00:50:56:95:68:c4 1 1 04/04 4002 A 11 0 0 1 1 0 0 0 0 0
3921 0 f07fea 00:50:56:95:24:8f 1 0 00/0f 8004 A d 0 0 1 0 0 0 0 1 0

```

Enter this command in order to view a list of the destination entries (look for the destination MAC address):

```

module-1# show platform internal ns forwarding gst-12
error opening file
: No such file or directory

```

```

=====
TABLE INSTANCE : 0
=====
Legend:
POS: Entry Position O: Overlay Instance
V: Valid Bit MD/PT: Mod/Port
PT: Pointer Type(A=Adj, E=ECMP, D=DstEncap N=Invalid)
PTR: ECMP/Adj/DstEncap/MET pointer
ML: MET Last
ST: Static PTH: Num Paths
BN: Bounce CP: Copy To CPU
PA: Policy Applied PI: Policy Incomplete
DL: Dst Local SP: Spine Proxy

```

```

-----

```

POS	O	VNID	Address	V	DE	MO	SRC	P	M	S	B	C	P	P	D	S		
						MD/PT	CLSS	T	L	T	PTH	N	P	A	I	L	P	
2139	0	ff7f72	00:50:56:95:7b:16	1	0	00/00	8006	A	d	0	0	1	0	0	0	0	1	0
2195	0	faff97	00:50:56:95:5d:6e	1	0	00/00	8005	A	f	0	0	1	0	0	0	0	1	0
3379	0	f07fea	00:50:56:95:bd:89	1	1	00/00	8004	A	10	0	0	1	0	0	0	0	0	0
4143	0	f07fea	00:50:56:95:17:b7	1	0	00/00	8004	A	a	0	0	1	0	0	0	0	1	0
4677	0	f07feb	00:50:56:95:68:c4	1	0	00/00	4002	A	e	0	0	1	0	0	0	0	1	0
5704	0	f07fea	00:50:56:95:24:8f	1	0	00/00	8004	A	a	0	0	1	0	0	0	0	1	0
6191	0	f7ffaf	00:50:56:95:00:33	1	0	00/00	4007	A	c	0	0	1	0	0	0	0	1	0

Take note of the Pointer (**PTR**) field in these outputs, which is the adjacency pointer. This value is used in the next command in order to find the destination encapsulated VLAN. This is a HEX value that you must convert to a decimal value (0 x 10 in decimal is 16).

Enter this command into the CLI, with **16** as the adjacency pointer:

```

module-1# show platform internal ns forwarding adj 16
error opening file
: No such file or directory

```

```

=====
TABLE INSTANCE : 0
=====
Legend
TD: TTL Dec Disable          UP: USE PCID
DM: Dst Mac Rewrite         SM: Src Mac Rewrite
RM IDX: Router Mac IDX      SR: Seg-ID Rewrite

```

```

-----

```

ENCP	T	U	USE	D	S	RM	S	SRC				
POS	SEG-ID	PTR	D	P	PCI	M	DST-MAC	M	IDX	R	SEG-ID	CLSS
16	0	2ffa	0	0	0	1	00:0c:0c:0c:0c:0c	0	0	0	0	0

Take note of the **ENCP PTR** value in this output, which is used in order to find the destination Tunnel

Endpoint (TEP) address:

```
module-1# show platform internal ns forwarding encap 0x2ffa
error opening file
: No such file or directory
```

```
=====
TABLE INSTANCE : 0
=====
Legend
MD: Mode (LUX & RWX)          LB: Loopback
LE: Loopback ECMP             LB-PT: Loopback Port
ML: MET Last                  TD: TTL Dec Disable
DV: Dst Valid                 DT-PT: Dest Port
DT-NP: Dest Port Not-PC      ET: Encap Type
OP: Override PIF Pinning     HR: Higig DstMod RW
HG-MD: Higig DstMode         KV: Keep VNTAG
=====
M  PORT  L  L  LB  MET  M  T  D  DT  DT  E  TST  O  H  HG  K  M  E
POS   D  FTAG B  E  PT  PTR  L  D  V  PT  NP  T  IDX  P  R  MD  V  D  T  Dst  MAC          DIP
=====
12282 0   c00 0 1   0   0 0 0 0 0 0 3   7 0 0   0 0 0 3 00:00:00:00:00:00 192.168.56.93
```

In this case, the frame is encapsulated in iVXLAN via the source IP address of the local TEP and destination IP address of the TEP that is listed. Based on the ELTMC output, the VXLAN ID for that BD is **15761386**, so this is the ID that is placed into the VXLAN packet. When the traffic reaches the other side, it is de-encapsulated, and since the destination MAC address is local, it is forwarded out of the port in the *l2 show* command from the Broadcom.

Single BD/Two EPGs with One Endpoint in Each EPG on the Same Leaf

This section describes how to verify the hardware programming and packet flow for two endpoints in different EPGs but with the same BD. The traffic flows to the same Leaf switch. This is also known as a Physical Local-to-Physical Local (PL-to-PL) Bridged packet. It is *Bridged* because communication is allowed between two encapsulated VLANs without the need for a Layer 3 (L3) interface to perform routing.

The first thing that you should verify is whether the MAC address information for both the source and destination IP addresses on the Leaf switches is learned on the expected interface (**1/48** in this case). This is the MAC and IP address information that is used in this example:

- Source MAC address: **0050.5695.908b**
- Source IP address: **192.168.1.50**
- Destination MAC address: **0050.5695.bd89**
- Destination IP address: **192.168.1.51**

Enter the *show mac address-table* command into the CLI in order to verify this information:

```
leaf1# show mac address-table | grep 908b
* 34      0050.5695.908b    dynamic    -      F      F      eth1/48
leaf1# show mac address-table | grep bd89
* 38      0050.5695.bd89    dynamic    -      F      F      eth1/48
```

You should then enter into the Broadcom (BCM) shell and verify that the BCM learns the correct MAC address information:

```
bcm-shell.0> l2 show
mac=00:50:56:95:bd:89 vlan=55 GPORT=0x30 modid=0 port=48/xe47
mac=00:50:56:95:90:8b vlan=54 GPORT=0x30 modid=0 port=48/xe47 Hit
```

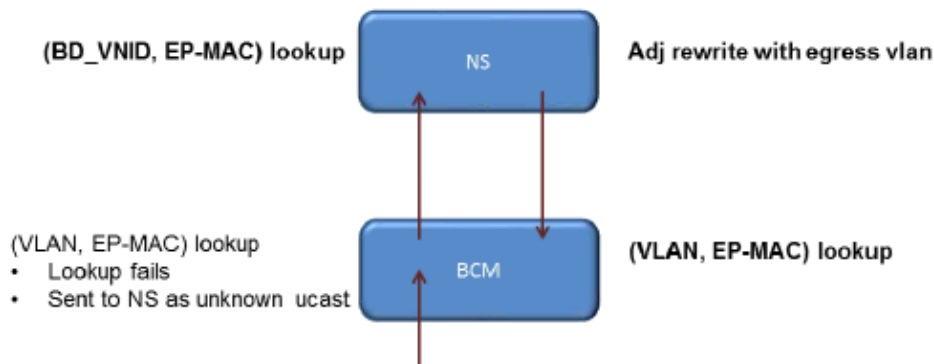
The output shows that the BCM has learned the MAC address information; however, the MAC addresses are on different VLANs. This is expected, as the traffic comes in from the host with different encapsulated VLANs (different EPGs).

Enter into the ELTMC in order verify the *HW_VlanID* that is displayed in the BCM shell against the BD VLAN for the two encapsulated VLANs:

```

module-1# show system internal eltmc info vlan brief
VLAN-Info
VlanId  HW_VlanId  Type                Access_enc  Access_enc  Fabric_enc  Fabric_enc  BDVlan
Type                Type
=====
13      15          BD_CTRL_VLAN       802.1q     4093        VXLAN       16777209    0
14      16          BD_VLAN            Unknown    0           VXLAN       15957970    14
15      17          BD_VLAN            Unknown    0           VXLAN       16613250    15
16      18          FD_VLAN            802.1q     301         VXLAN       8509        15
17      19          BD_VLAN            Unknown    0           VXLAN       16220082    17
18      46          BD_VLAN            Unknown    0           VXLAN       14745592    18
19      50          BD_VLAN            Unknown    0           VXLAN       16646015    19
20      51          FD_VLAN            802.1q     502         VXLAN       8794        19
21      23          BD_VLAN            Unknown    0           VXLAN       16121792    21
22      24          FD_VLAN            802.1q     538         VXLAN       8830        21
23      25          BD_VLAN            Unknown    0           VXLAN       15826915    23
24      28          FD_VLAN            802.1q     537         VXLAN       8829        23
25      26          BD_VLAN            Unknown    0           VXLAN       16351138    25
26      29          FD_VLAN            802.1q     500         VXLAN       8792        25
27      27          BD_VLAN            Unknown    0           VXLAN       16678779    27
28      30          FD_VLAN            802.1q     534         VXLAN       8826        27
29      52          BD_VLAN            Unknown    0           VXLAN       15859681    29
31      47          FD_VLAN            802.1q     602         VXLAN       9194        18
32      31          FD_VLAN            802.1q     292         VXLAN       8500        55
33      20          BD_VLAN            Unknown    0           VXLAN       15761386    33
34      54          FD_VLAN            802.1q     299         VXLAN       8507        54
35      33          BD_VLAN            Unknown    0           VXLAN       16449431    35
38      55          FD_VLAN            802.1q     300         VXLAN       8508        54
39      53          FD_VLAN            802.1q     501         VXLAN       8793        29
  
```

In this ELTMC output, you can see that the *HW_VlanId* for each entry is mapped to the *Access_enc* that the traffic is tagged with when it enters the switch (check the VMware port groups in order to verify whether it is virtualized) and that the *VlanId* is the PI VLAN that appeared in the MAC address table. This is a Bridged connection in this case because the BD VLAN is the same (they are both on VLAN 54). This diagram shows the BCM-to-NorthStar interaction:



NorthStar adjusts the packet and rewrites the egress frame with the *HW_VlanId* of the destination IP address. This way, the BCM has a local hit in that VLAN and sends the frame out through port *1/48*.

Two BDs/Two EPGs with One Endpoint in Each EPG on the Same Leaf (Routed Packet)

This section describes how to verify the hardware programming and packet flow for two endpoints in different EPGs that use different BDs. The traffic flows to the same Leaf switch, but it must be routed. This is also known as a PL-to-PL *Routed* packet.

The first thing that you should verify is whether the MAC address information for both the source and destination IP addresses on the Leaf switch is learned on the expected interface (*I/48* in this case). This is the MAC and IP address information that is used in this example:

- Source MAC address: **0050.5695.908b**
- Source IP address: **192.168.1.50**
- Default Gateway: **192.168.1.1**
- Destination MAC address: **0050.5695.bd89**
- Destination IP address: **192.168.3.51**
- Default Gateway: **192.168.3.1**

While you can view the MAC address table in order to verify the L2 information, an important piece of the solution for the L3 routed traffic is the Endpoint Manager (EPM). The EPM is the process that tracks all of the endpoints on a particular device.

Verify that the EPM has knowledge of the two endpoints on the first Leaf switch (*Leaf1*):

```
leaf1# show endpoint ip 192.168.1.50
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   |           |
+-----+-----+-----+-----+-----+
56      |      | 0050.5695.908b L |          | eth1/48  |
Joey-Tenant:Joey-Internal |      | 192.168.1.50 L  |          |          |
```

The source IP address is learned on Ethernet *I/48*, and it is local to this switch.

```
leaf1# show endpoint ip 192.168.3.51
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   |           |
+-----+-----+-----+-----+-----+
44      |      | 0050.5695.bd89 L |          | eth1/48  |
Joey-Tenant:Joey-Internal |      | 192.168.3.51 L  |          |          |
```

As shown, the destination IP address is learned on Ethernet *I/48* and it is local to this switch.

In order to obtain more detailed information about these endpoints, connect to the Linecard (LC):

```
leaf1# vsh_lc
module-1# show system internal epmc endpoint ip 192.168.1.50

MAC : 0050.5695.908b ::: Num IPs : 1
IP# 0 : 192.168.1.50 ::: IP# 0 flags :
```



```

Vlan id : 56 ::: Vlan vnid : 8507 ::: BD vnid : 15990734
VRF vnid : 2523136 ::: phy if : 0x1a02f000 ::: tunnel if : 0
Interface : Ethernet1/48
VTEP tunnel if : N/A ::: Flags : 0x80004c04
Ref count : 5 ::: sclass : 0x2ab5
Timestamp : 02/01/1970 00:43:53.129731
last mv timestamp 12/31/1969 19:00:00.000000 ::: ep move count : 0
previous if : 0 ::: loop detection count : 0
EP Flags : local,IP,MAC,class-set,timer,
Aging:Timer-type : Host-tracker timeout ::: Timeout-left : 423 ::: Hit-bit :
Yes ::: Timer-reset count : 406

PD handles:
Bcm l2 hit-bit : Yes
[L2]: Asic : NS ::: ADJ : 0x14 ::: LST SA : 0x83a ::: LST DA : 0x83a :::
GST ING : 0xedb ::: BCM : Yes
[L3-0]: Asic : NS ::: ADJ : 0x14 ::: LST SA : 0xe56 ::: LST DA : 0xe56 :::
GST ING : 0x12ae ::: BCM : Yes
:::

```

Take note of the *VRF vnid* and the *BD vnid* values.

```

module-1# show system internal epmc endpoint ip 192.168.3.51

MAC : 0050.5695.bd89 ::: Num IPs : 1
IP# 0 : 192.168.3.51 ::: IP# 0 flags :
Vlan id : 44 ::: Vlan vnid : 8499 ::: BD vnid : 15761386
VRF vnid : 2523136 ::: phy if : 0x1a02f000 ::: tunnel if : 0
Interface : Ethernet1/48
VTEP tunnel if : N/A ::: Flags : 0x80004c04
Ref count : 5 ::: sclass : 0x8004
Timestamp : 02/01/1970 00:43:53.130524
last mv timestamp 12/31/1969 19:00:00.000000 ::: ep move count : 0
previous if : 0 ::: loop detection count : 0
EP Flags : local,IP,MAC,class-set,timer,
Aging:Timer-type : Host-tracker timeout ::: Timeout-left : 532 ::: Hit-bit :
Yes ::: Timer-reset count : 1

PD handles:
Bcm l2 hit-bit : Yes
[L2]: Asic : NS ::: ADJ : 0x15 ::: LST SA : 0x28e ::: LST DA : 0x28e :::
GST ING : 0xd33 ::: BCM : Yes
[L3-0]: Asic : NS ::: ADJ : 0x15 ::: LST SA : 0x497b ::: LST DA : 0x497b :::
GST ING : 0x1e98 ::: BCM : Yes
:::

```

The *VRF vnid* value in this output is the same because both of the routes are a part of the same Virtual Routing and Forwarding (VRF) in the routing table (same context). The *BD vnid* value is different, since the two endpoints are in different BDs.

Just as you viewed the NorthStar tables in order to verify the hardware programming for the MAC addresses at an L2 level, you can do the same in order to verify the L3 table:

```

module-1# show platform internal ns forwarding lst-13
error opening file
: No such file or directory

=====
TABLE INSTANCE : 0
=====
Legend:
POS: Entry Position          O: Overlay Instance
V: Valid Bit                 MD/PT: Mod/Port
PT: Pointer Type(A=Adj, E=ECMP, D=DstEncap N=Invalid)

```

PTR: ECMP/Adj/DstEncap/MET pointer
 ML: MET Last
 ST: Static
 BN: Bounce
 PA: Policy Applied
 DL: Dst Local
 PTH: Num Paths
 CP: Copy To CPU
 PI: Policy Incomplete
 SP: Spine Proxy

MO	SRC	P	M	S	B	C	P	P	D	S										
POS	O	VNID	Address		V	DE	MD/PT	CLSS	T	PTR	L	T	PTH	N	P	A	I	L	P	
2881	0	268000	192.168.1.1		1	0	00/00	1	A	0	0	1	1	0	0	0	1	0	0	
3003	0	208001	80.80.80.10		1	0	00/14	800d	A	0	0	0	1	0	0	0	1	0	0	
3051	0	208001	30.30.30.30		1	0	00/14	c009	A	0	0	0	1	0	0	0	0	0	0	
3328	0	268000	192.168.2.1		1	0	00/00	1	A	0	0	1	1	0	0	0	1	0	0	
3670	0	268000	192.168.1.50		1	0	00/09	2ab5	A	0	0	0	1	0	0	0	0	0	0	
3721	0	2b8001	50.50.50.1		1	0	00/00	1	A	0	0	1	1	0	0	0	1	0	0	
3903	0	268000	192.168.3.1		1	0	00/00	1	A	0	0	1	1	0	0	0	1	0	0	
18811	0	268000	192.168.3.51		1	0	00/09	8004	A	0	0	0	1	0	0	0	0	0	0	

This diagram illustrates the flow through the ASICs:

