# Troubleshoot ACI Intra-Fabric Forwarding - MultiPod Forwarding

## Contents

[Other references]

# Introduction

This document describes steps to understand and troubleshoot an ACI Multi-Pod Forwarding Scenario.

# Background Information

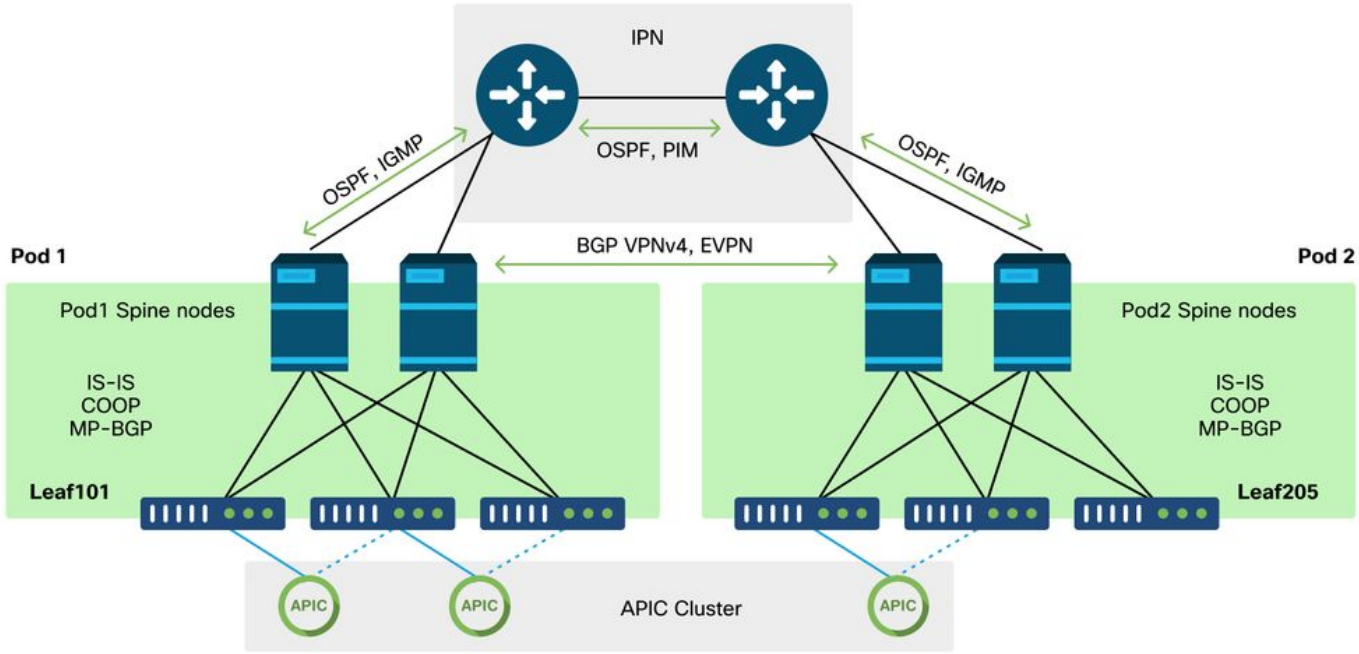
The material from this document was extracted from the [Troubleshooting Cisco Application Centric Infrastructure, Second Edition] book, specifically the **Intra-Fabric fowarding - Multi-Pod Forwarding** chapter.

# Multi-Pod Forwarding Overview

This chapter will cover how to troubleshoot scenarios in which connectivity is not working correctly across Pods in a Multi-Pod environment

Before looking at specific troubleshooting examples, it is important to take a moment to understand the Multi-Pod components at a high level.

# Multi-Pod Components




Similar to a traditional ACI fabric, a Multi-Pod fabric is still considered to be a single ACI fabric and relies on a single APIC cluster for management.

Within each individual Pod, ACI leverages the same protocols in the overlay as a traditional fabric. This includes IS-IS for exchange of TEP information as well as multicast Outgoing Interface (OIF) selection, COOP for a global endpoint repository, and BGP VPNv4 for the distribution of external routers through the fabric.

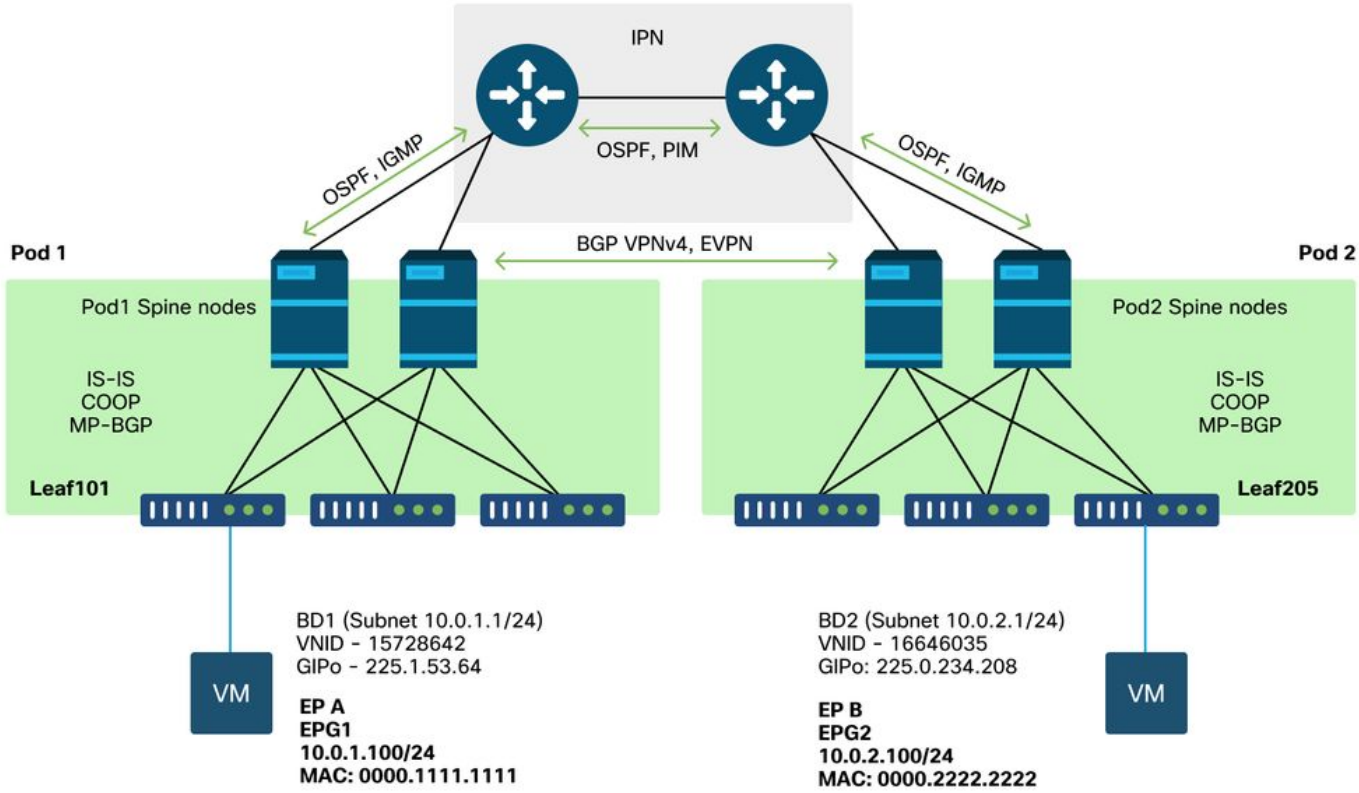
Multi-Pod builds on those components as it must connect each Pod together.

- To exchange routing information regarding TEPs in the remote Pod, OSPF is used to advertise the summary TEP pool through the IPN.
- To exchange external routes learned from one Pod to another, the BGP VPNv4 address-family is extended between spine nodes. Each Pod becomes a separate route-reflector cluster.
- To synchronize endpoints as well as other information stored in COOP across Pods, the BGP EVPN address-family is extended between spine nodes.
- Lastly, in order to handle the flooding of Broadcast, Unknown-Unicast, and Multicast (BUM) traffic across Pods, the spine nodes in each Pod act as IGMP hosts and the IPN routers exchange multicast routing information through Bidirectional PIM.

A large portion of the Multi-Pod troubleshooting scenarios and workflows are similar to Single Pod ACI fabrics. This Multi-Pod section will mostly focus on the differences between Single Pod and Multi-Pod forwarding.

## Topology for Multi-Pod Examples

As with troubleshooting any scenario, it is important to begin by understanding what the expected state is. Reference this topology for this chapter's examples.




## General Workflow for Troubleshooting Multi-Pod Forwarding

At a high level, when debugging a multi-pod forwarding issue, the following steps can be evaluated:

1. Is the flow Unicast or multi-destination? Remember, even if the flow is expected to be unicast in the working state, if ARP isn't resolved then it is a multi-destination flow.
2. Is the flow routed or bridged? Traditionally, a routed flow from an ACI perspective would be any flow where the destination MAC address is the router MAC address that is owned by a gateway configured on ACI. Additionally, if ARP flooding is disabled, then the ingress leaf would route based on the target-IP address. If the destination MAC address is not owned by ACI, then the switch would either forward based on the MAC address or follow the 'unknown unicast' behavior configured on the bridge domain.
3. Is the ingress leaf dropping the flow? fTriage and ELAM are the best tools to confirm this.

**If the flow is Layer 3 unicast:**

1. Does the ingress leaf have an endpoint learn for the destination IP in the same VRF as the source EPG? If so, this will always take precedence over any learned routes. The leaf will forward directly to the tunnel address or egress interface where the endpoint is learned.
2. If there is no endpoint learn, does the ingress leaf have a route for the destination that has the 'Pervasive' flag set? This indicates that the destination subnet is configured as a Bridge Domain subnet and that the next-hop should be the spine proxy in the local Pod.
3. If there is no Pervasive route, then the last resort would be any routes that are learned through an L3Out. This portion is identical to Single Pod L3Out forwarding.

**If the flow is Layer 2 unicast:**

1. Does the ingress leaf have an endpoint learn for the destination MAC address in the same Bridge Domain as the source EPG? If so, the leaf will forward to the remote tunnel IP or out the local interface where the endpoint is learned.
2. If there is no learn for the destination MAC address in the source Bridge Domain, then the leaf will forward based on the 'unknown-unicast' behavior the BD is set to. If it is set to 'Flood', then the leaf will flood to the GIPo multicast group allocated for the Bridge Domain. Local and remote Pods should get a flooded copy. If it is set to 'Hardware Proxy' then the frame is sent to the spine for a proxy lookup and forwarded based on the spine's COOP entry.

Since the troubleshooting outputs would be considerably different for unicast compared to BUM, working outputs and scenarios for unicast will be considered before and then move to BUM.

# Multi-Pod Unicast Troubleshooting Workflow

Following the topology, walk through the flow from 10.0.2.100 on leaf205 to 10.0.1.100 on leaf101.

Note, before proceeding here, it is important to confirm whether the source has ARP resolved for the gateway (for a routed flow) or the destination MAC address (for a bridged flow)

### 1. Confirm that the ingress leaf receives the packet. Use the ELAM CLI tool shown in the "Tools" section along with the ereport output available in 4.2. The ELAM Assistant App is also used.

```
module-1# debug platform internal tah elam asic 0
module-1(DBG-elam)# trigger reset
module-1(DBG-elam)# trigger init in-select 6 out-select 1
```

```
module-1(DBG-elam-insel6)# set outer ipv4 src_ip 10.0.2.100 dst_ip 10.0.1.100
module-1(DBG-elam-insel6)# start
module-1(DBG-elam-insel6)# status
ELAM STATUS
===========
Asic 0 Slice 0 Status Armed
Asic 0 Slice 1 Status Triggered
```

Note that the ELAM triggered which confirms the packet was received on the ingress switch. Now look at a couple of fields in the report since the output is extensive.

```
================================================================================================
===========
                                                        Captured Packet

================================================================================================
===========

------------------------------------------------------------------------------------------------
-----------
Outer Packet Attributes
------------------------------------------------------------------------------------------------
-----------
Outer Packet Attributes       : l2uc ipv4 ip ipuc ipv4uc
Opcode                        : OPCODE_UC

------------------------------------------------------------------------------------------------
-----------
Outer L2 Header
------------------------------------------------------------------------------------------------
-----------
Destination MAC               : 0022.BDF8.19FF
Source MAC                    : 0000.2222.2222
802.1Q tag is valid           : yes( 0x1 )
CoS                           : 0( 0x0 )
Access Encap VLAN             : 1021( 0x3FD )

------------------------------------------------------------------------------------------------
------------
Outer L3 Header
------------------------------------------------------------------------------------------------
------------
L3 Type                       : IPv4
IP Version                    : 4
DSCP                          : 0
IP Packet Length              : 84 ( = IP header(28 bytes) + IP payload )
Don't Fragment Bit            : not set
TTL                           : 255
IP Protocol Number            : ICMP
IP CheckSum                   : 10988( 0x2AEC )
Destination IP                : 10.0.1.100
Source IP                     : 10.0.2.100
```

There is much more info in the ereport about where the packet is going but the ELAM Assistant App is currently more useful for interpreting this data. The ELAM Assistant output for this flow will be shown later in this chapter.

## 2. Is the ingress leaf learning the destination as an endpoint in the ingress VRF? If not, is there a route?

```
a-leaf205# show endpoint ip 10.0.1.100 detail
Legend:
s - arp              H - vtep             V - vpc-attached     p - peer-aged
R - peer-attached-rl B - bounce           S - static           M - span
D - bounce-to-proxy  O - peer-attached    a - local-aged       m - svc-mgr
L - local            E - shared-service
+------------------------------------+---------------+-----------------+--------------+----------
---+------------------------------+
    VLAN/                           Encap            MAC Address       MAC Info/
Interface     Endpoint Group
    Domain                          VLAN             IP Address        IP Info
                  Info
+------------------------------------+---------------+-----------------+--------------+----------
---+------------------------------+
```

No output in the above command means the destination IP is not learned. Next check the routing table.

```
a-leaf205# show ip route 10.0.1.100 vrf Prod:Vrf1
IP Route Table for VRF "Prod:Vrf1"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

10.0.1.0/24, ubest/mbest: 1/0, attached, direct, pervasive
  *via 10.0.120.34%overlay-1, [1/0], 01:55:37, static, tag 4294967294
       recursive next hop: 10.0.120.34/32%overlay-1
```

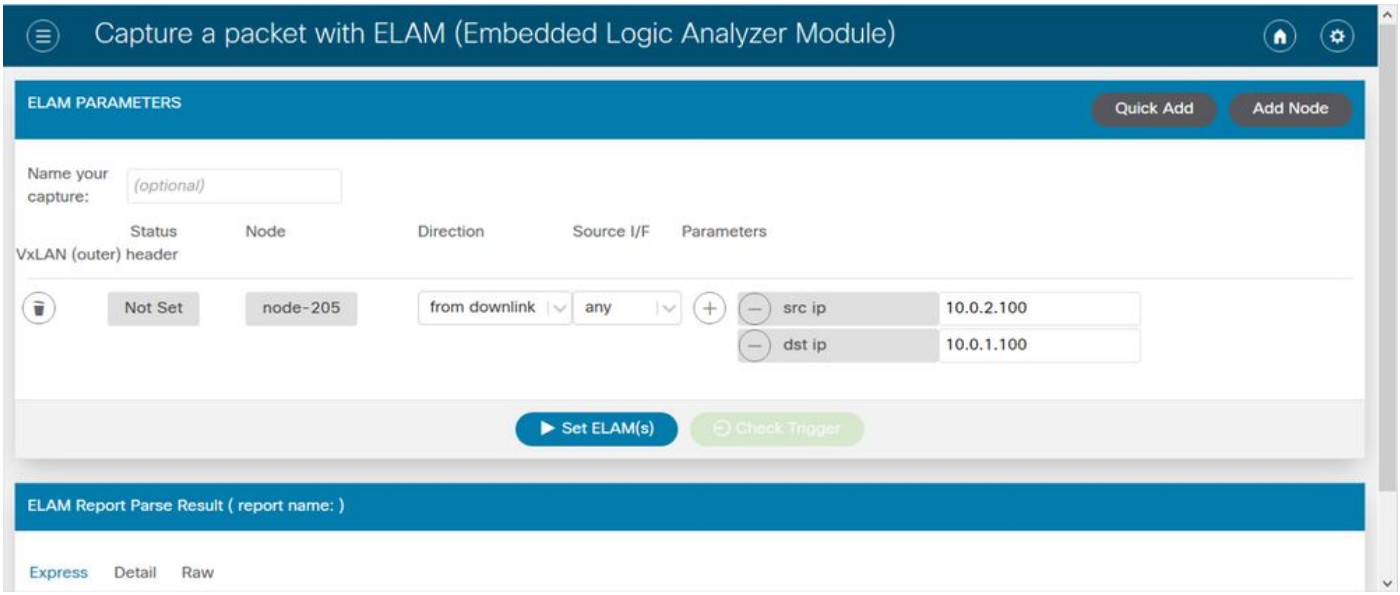
In the above output, the pervasive flag is seen which indicates this is a Bridge Domain subnet route. The next-hop should be an anycast proxy address on the spines.

```
a-leaf205# show isis dtep vrf overlay-1 | grep 10.0.120.34
10.0.120.34        SPINE   N/A              PHYSICAL,PROXY-ACAST-V4
```

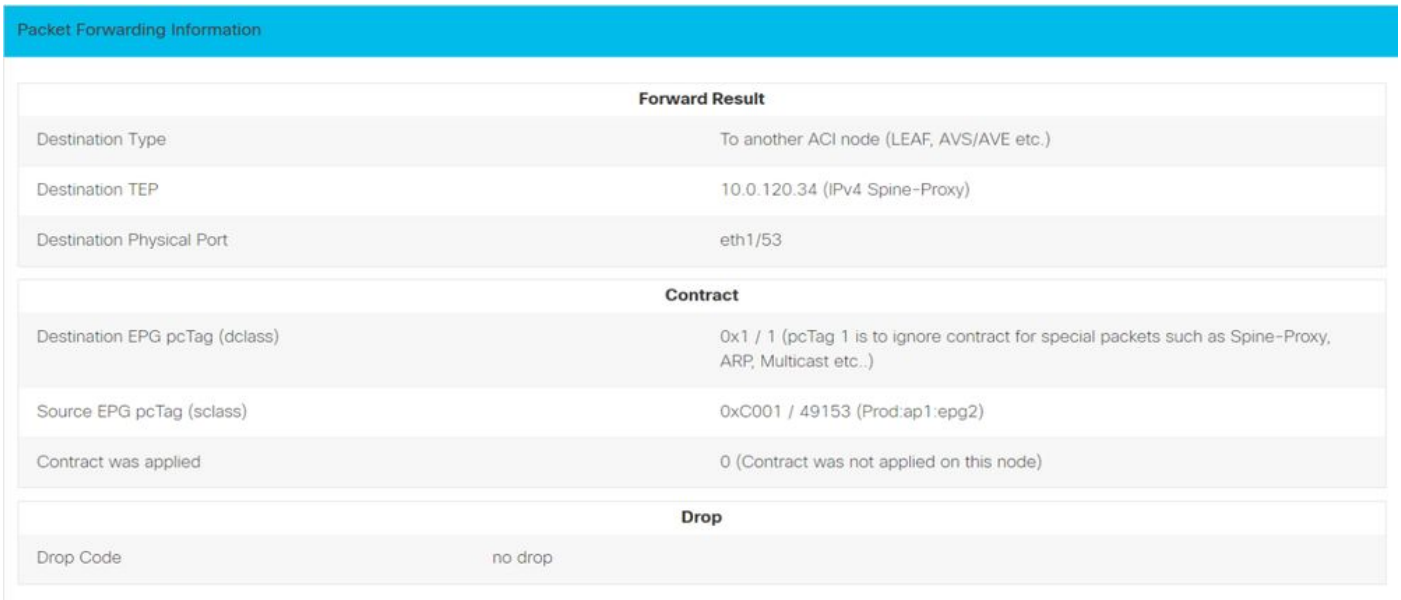
Note that if the endpoint is learned on a tunnel or physical interface, this will take precedence, causing the packet to be forwarded directly there. Refer to the "External forwarding" chapter of this book for more details.

Use the ELAM Assistant to confirm the forwarding decisions seen in the above outputs.

### ELAM Assistant Configuration




### Verify Forwarding Decisions




The above output shows that the ingress leaf is forwarding the packet to the IPv4 spine proxy address. This is what is expected to happen.

## 3. Confirm on the spine that the destination IP is present in COOP so that the proxy request works.

There are multiple ways to get the COOP output on the spine, for example, look at it with a 'show coop internal info ip-db' command:

```
a-spine4# show coop internal info ip-db | grep -B 2 -A 15 "10.0.1.100"

------------------------------
IP address : 10.0.1.100
Vrf : 2392068 <-- This vnid should correspond to vrf where the IP is learned. Check operational
tab of the tenant vrfs
Flags : 0x2
EP bd vnid : 15728642
EP mac :  00:00:11:11:11:11
```

```
Publisher Id : 192.168.1.254
Record timestamp : 12 31 1969 19:00:00 0
Publish timestamp : 12 31 1969 19:00:00 0
Seq No: 0
Remote publish timestamp: 09 30 2019 20:29:07 9900483
URIB Tunnel Info
Num tunnels : 1
      Tunnel address : 10.0.0.34 <-- When learned from a remote pod this will be an External
Proxy TEP. We'll cover this more
      Tunnel ref count : 1

------------------------------
```

Other commands to run on the spine:

**Query COOP for l2 entry:**

```
moquery -c coopEpRec -f 'coop.EpRec.mac=="00:00:11:11:22:22"
```

**Query COOP for l3 entry and get parent l2 entry:**

```
moquery -c coopEpRec -x rsp-subtree=children 'rsp-subtree-
filter=eq(coopIpv4Rec.addr,"192.168.1.1")' rsp-subtree-include=required
```

**Query COOP for l3 entry only:**

```
moquery -c coopIpv4Rec -f 'coop.Ipv4Rec.addr=="192.168.1.1"'
```

The useful thing about the multiple moquery is that they can also be run directly on an APIC and the user can see every spine that has the record in coop.

## 4. Multi-Pod spine proxy forwarding decision

If the spine's COOP entry points to a tunnel in the local Pod then forwarding is based on traditional ACI behavior.

Note that owner of a TEP can be verified in the fabric by running from an APIC: **moquery -c ipv4Addr -f 'ipv4.Addr.addr=="<tunnel address>"'**

In the proxy scenario, the tunnel next-hop is 10.0.0.34. Who is the owner of this IP address?:

```
a-apic1# moquery -c ipv4Addr -f 'ipv4.Addr.addr=="10.0.0.34"' | grep dn
dn               : topology/pod-1/node-1002/sys/ipv4/inst/dom-overlay-1/if-[lo9]/addr-
[10.0.0.34/32]
dn               : topology/pod-1/node-1001/sys/ipv4/inst/dom-overlay-1/if-[lo2]/addr-
[10.0.0.34/32]
```

This IP is owned by both spine nodes in Pod 1. This is a specific IP called an External Proxy address. In the same way that ACI has proxy addresses owned by the spine nodes within a Pod (see step 2 of this section), there are also proxy addresses assigned to the Pod itself. This interface type can be verified by running:

```
a-apic1# moquery -c ipv4If -x rsp-subtree=children 'rsp-subtree-
filter=eq(ipv4Addr.addr,"10.0.0.34")' rsp-subtree-include=required

...
```

```
# ipv4.If
mode         : anycast-v4,external

# ipv4.Addr
 addr             : 10.0.0.34/32
 dn               : topology/pod-1/node-1002/sys/ipv4/inst/dom-overlay-1/if-[lo9]/addr-
[10.0.0.34/32]
```

The 'external' flag indicates this is an external proxy TEP.

## 5. Verify BGP EVPN on the spine

The coop endpoint record should be imported from BGP EVPN on the spine. The following command can be used to verify that it is in EVPN (though if it is already in COOP with a next-hop of the remote Pod external proxy TEP it can be assumed it came from EVPN):

```
a-spine4# show bgp l2vpn evpn 10.0.1.100 vrf overlay-1
Route Distinguisher: 1:16777199
BGP routing table entry for [2]:[0]:[15728642]:[48]:[0000.1111.1111]:[32]:[10.0.1.100]/272,
version 689242 dest ptr 0xaf42a4ca
Paths: (2 available, best #2)
Flags: (0x000202 00000000) on xmit-list, is not in rib/evpn, is not in HW, is locked
Multipath: eBGP iBGP

 Path type: internal 0x40000018 0x2040 ref 0 adv path ref 0, path is valid, not best reason:
Router Id, remote nh not installed
 AS-Path: NONE, path sourced internal to AS
  192.168.1.254 (metric 7) from 192.168.1.102 (192.168.1.102)
    Origin IGP, MED not set, localpref 100, weight 0
    Received label 15728642 2392068
    Received path-id 1
    Extcommunity:
        RT:5:16
        SOO:1:1
        ENCAP:8
        Router MAC:0200.0000.0000

          Advertised path-id 1
 Path type: internal 0x40000018 0x2040 ref 1 adv path ref 1, path is valid, is best path, remote
nh not installed
 AS-Path: NONE, path sourced internal to AS
  192.168.1.254 (metric 7) from 192.168.1.101 (192.168.1.101)
    Origin IGP, MED not set, localpref 100, weight 0
    Received label 15728642 2392068
    Received path-id 1
    Extcommunity:
        RT:5:16
        SOO:1:1
        ENCAP:8
        Router MAC:0200.0000.0000

          Path-id 1 not advertised to any peer
```

Note that the above command can be run for a MAC address as well.

-192.168.1.254 is the dataplane TEP configured during Multi-Pod setup. Note however that even though it is advertised in BGP as the NH, the actual next-hop will be the external proxy TEP.

-192.168.1.101 and .102 are the Pod 1 spine nodes advertising this path.

## 6. Verify COOP on the spines in the destination Pod.

The same command as earlier can be used:

```
a-spine2# show coop internal info ip-db | grep -B 2 -A 15 "10.0.1.100"

------------------------------
IP address : 10.0.1.100
Vrf : 2392068
Flags : 0
EP bd vnid : 15728642
EP mac :  00:50:56:81:3E:E6
Publisher Id : 10.0.72.67
Record timestamp : 10 01 2019 15:46:24 502206158
Publish timestamp : 10 01 2019 15:46:24 524378376
Seq No: 0
Remote publish timestamp: 12 31 1969 19:00:00 0
URIB Tunnel Info
Num tunnels : 1
      Tunnel address : 10.0.72.67
      Tunnel ref count : 1

------------------------------
```

Verify who owns the tunnel address by running the following command on an APIC:

```
a-apic1# moquery -c ipv4Addr -f 'ipv4.Addr.addr=="10.0.72.67"'
Total Objects shown: 1

# ipv4.Addr
addr             : 10.0.72.67/32
childAction      :
ctrl             :
dn               : topology/pod-1/node-101/sys/ipv4/inst/dom-overlay-1/if-[lo0]/addr-
[10.0.72.67/32]
ipv4CfgFailedBmp :
ipv4CfgFailedTs  : 00:00:00:00.000
ipv4CfgState     : 0
lcOwn            : local
modTs            : 2019-09-30T18:42:43.262-04:00
monPolDn         : uni/fabric/monfab-default
operSt           : up
operStQual       : up
pref             : 0
rn               : addr-[10.0.72.67/32]
status           :
tag              : 0
type             : primary
vpcPeer          : 0.0.0.0
```

The above command shows that the tunnel from COOP points to leaf101. This means that leaf101 should have the local learn for the destination endpoint.

## 7. Verify that the egress leaf has the local learn.

This can be done via a 'show endpoint' command:

```
a-leaf101# show endpoint ip 10.0.1.100 detail
```

```
Legend:
s - arp              H - vtep             V - vpc-attached     p - peer-aged
R - peer-attached-rl B - bounce           S - static           M - span
D - bounce-to-proxy  O - peer-attached    a - local-aged       m - svc-mgr
L - local            E - shared-service
+-----------------------------------+--------------+-----------------+-------------+----------
---+------------------------------+
    VLAN/                           Encap           MAC Address       MAC Info/
Interface     Endpoint Group
    Domain                          VLAN            IP Address        IP
Info                       Info
+-----------------------------------+--------------+-----------------+-------------+----------
---+------------------------------+
341                                   vlan-1075    0000.1111.1111 LV
po5                  Prod:ap1:epg1
Prod:Vrf1                             vlan-1075        10.0.1.100 LV
po5
```

Note that the endpoint is learned. The packet should be forwarded based out port-channel 5 with VLAN tag 1075 set.

## Using fTriage to verify the end-to-end flow

As discussed in the "Tools" section of this chapter, fTriage can be used to map out an existing flow end-to-end and understand what every switch in the path is doing with the packet. This is particularly useful in larger and more complex deployments such as Multi-Pod.

Note that fTriage will take some time to fully run (potentially 15 minutes).

When running fTriage on the example flow:

```
a-apic1# ftriage route -ii LEAF:205 -dip 10.0.1.100 -sip 10.0.2.100
fTriage Status: {"dbgFtriage": {"attributes": {"operState": "InProgress", "pid": "7297",
"apicId": "1", "id": "0"}}}
Starting ftriage
Log file name for the current run is: ftlog_2019-10-01-16-04-15-438.txt
2019-10-01 16:04:15,442 INFO     /controller/bin/ftriage route -ii LEAF:205 -dip 10.0.1.100 -sip
10.0.2.100
2019-10-01 16:04:38,883 INFO     ftriage:     main:1165 Invoking ftriage with default password
and default username: apic#fallback\\admin
2019-10-01 16:04:54,678 INFO     ftriage:     main:839  L3 packet Seen on a-leaf205 Ingress:
Eth1/31  Egress: Eth1/53 Vnid: 2392068
2019-10-01 16:04:54,896 INFO     ftriage:     main:242  ingress encap string vlan-1021
2019-10-01 16:04:54,899 INFO     ftriage:     main:271  Building ingress BD(s), Ctx
2019-10-01 16:04:56,778 INFO     ftriage:     main:294  Ingress BD(s) Prod:Bd2
2019-10-01 16:04:56,778 INFO     ftriage:     main:301  Ingress Ctx: Prod:Vrf1
2019-10-01 16:04:56,887 INFO     ftriage:   pktrec:490  a-leaf205: Collecting transient losses
snapshot for LC module: 1
2019-10-01 16:05:22,458 INFO     ftriage:     main:933  SIP 10.0.2.100 DIP 10.0.1.100
2019-10-01 16:05:22,459 INFO     ftriage:  unicast:973  a-leaf205: <- is ingress node
2019-10-01 16:05:25,206 INFO     ftriage:  unicast:1215 a-leaf205: Dst EP is remote
2019-10-01 16:05:26,758 INFO     ftriage:     misc:657  a-leaf205: DMAC(00:22:BD:F8:19:FF) same
as RMAC(00:22:BD:F8:19:FF)
2019-10-01 16:05:26,758 INFO     ftriage:     misc:659  a-leaf205: L3 packet getting
routed/bounced in SUG
2019-10-01 16:05:27,030 INFO     ftriage:     misc:657  a-leaf205: Dst IP is present in SUG L3
tbl
2019-10-01 16:05:27,473 INFO     ftriage:     misc:657  a-leaf205: RwDMAC DIPo(10.0.72.67) is
one of dst TEPs ['10.0.72.67']
2019-10-01 16:06:25,200 INFO     ftriage:     main:622  Found peer-node a-spine3 and IF: Eth1/31
```

```
in candidate list
2019-10-01 16:06:30,802 INFO     ftriage:     node:643  a-spine3: Extracted Internal-port GPD
Info for lc: 1
2019-10-01 16:06:30,803 INFO     ftriage:     fcls:4414 a-spine3: LC trigger ELAM with IFS:
Eth1/31 Asic :3 Slice: 1 Srcid: 24
2019-10-01 16:07:05,717 INFO     ftriage:     main:839  L3 packet Seen on a-spine3 Ingress:
Eth1/31 Egress: LC-1/3 FC-24/0 Port-1 Vnid: 2392068
2019-10-01 16:07:05,718 INFO     ftriage:   pktrec:490  a-spine3: Collecting transient losses
snapshot for LC module: 1
2019-10-01 16:07:28,043 INFO     ftriage:      fib:332  a-spine3: Transit in spine
2019-10-01 16:07:35,902 INFO     ftriage:  unicast:1252 a-spine3: Enter dbg_sub_nexthop with
Transit inst: ig infra: False glbs.dipo: 10.0.72.67
2019-10-01 16:07:36,018 INFO     ftriage:  unicast:1417 a-spine3: EP is known in COOP (DIPo =
10.0.72.67)
2019-10-01 16:07:40,422 INFO     ftriage:  unicast:1458 a-spine3: Infra route 10.0.72.67 present
in RIB
2019-10-01 16:07:40,423 INFO     ftriage:     node:1331 a-spine3: Mapped LC interface: LC-1/3
FC-24/0 Port-1 to FC interface: FC-24/0 LC-1/3 Port-1
2019-10-01 16:07:46,059 INFO     ftriage:     node:460  a-spine3: Extracted GPD Info for fc: 24
2019-10-01 16:07:46,060 INFO     ftriage:     fcls:5748 a-spine3: FC trigger ELAM with IFS: FC-
24/0 LC-1/3 Port-1 Asic :0 Slice: 1 Srcid: 40
2019-10-01 16:08:06,735 INFO     ftriage:  unicast:1774 L3 packet Seen on FC of node: a-spine3
with Ingress: FC-24/0 LC-1/3 Port-1  Egress: FC-24/0 LC-1/3 Port-1 Vnid: 2392068
2019-10-01 16:08:06,735 INFO     ftriage:   pktrec:487  a-spine3: Collecting transient losses
snapshot for FC module: 24
2019-10-01 16:08:09,123 INFO     ftriage:     node:1339 a-spine3: Mapped FC interface: FC-24/0
LC-1/3 Port-1 to LC interface: LC-1/3 FC-24/0 Port-1
2019-10-01 16:08:09,124 INFO     ftriage:  unicast:1474 a-spine3: Capturing Spine  Transit pkt-
type L3 packet on egress LC on Node: a-spine3 IFS: LC-1/3 FC-24/0 Port-1
2019-10-01 16:08:09,594 INFO     ftriage:     fcls:4414 a-spine3: LC trigger ELAM with IFS: LC-
1/3 FC-24/0 Port-1 Asic :3 Slice: 1 Srcid: 48
2019-10-01 16:08:44,447 INFO     ftriage:  unicast:1510 a-spine3: L3 packet Spine egress
Transit pkt Seen on a-spine3 Ingress: LC-1/3 FC-24/0 Port-1 Egress: Eth1/29 Vnid: 2392068
2019-10-01 16:08:44,448 INFO     ftriage:   pktrec:490  a-spine3: Collecting transient losses
snapshot for LC module: 1
2019-10-01 16:08:46,691 INFO     ftriage:  unicast:1681 a-spine3: Packet is exiting the fabric
through {a-spine3: ['Eth1/29']} Dipo 10.0.72.67 and filter SIP 10.0.2.100 DIP 10.0.1.100
2019-10-01 16:10:19,947 INFO     ftriage:     main:716  Capturing L3 packet Fex: False on node:
a-spine1 IF: Eth2/25
2019-10-01 16:10:25,752 INFO     ftriage:     node:643  a-spine1: Extracted Internal-port GPD
Info for lc: 2
2019-10-01 16:10:25,754 INFO     ftriage:     fcls:4414 a-spine1: LC trigger ELAM with IFS:
Eth2/25 Asic :3 Slice: 0 Srcid: 24
2019-10-01 16:10:51,164 INFO     ftriage:     main:716  Capturing L3 packet Fex: False on node:
a-spine2 IF: Eth1/31
2019-10-01 16:11:09,690 INFO     ftriage:     main:839  L3 packet Seen on a-spine2 Ingress:
Eth1/31 Egress: Eth1/25 Vnid: 2392068
2019-10-01 16:11:09,690 INFO     ftriage:   pktrec:490  a-spine2: Collecting transient losses
snapshot for LC module: 1
2019-10-01 16:11:24,882 INFO     ftriage:      fib:332  a-spine2: Transit in spine
2019-10-01 16:11:32,598 INFO     ftriage:  unicast:1252 a-spine2: Enter dbg_sub_nexthop with
Transit inst: ig infra: False glbs.dipo: 10.0.72.67
2019-10-01 16:11:32,714 INFO     ftriage:  unicast:1417 a-spine2: EP is known in COOP (DIPo =
10.0.72.67)
2019-10-01 16:11:36,901 INFO     ftriage:  unicast:1458 a-spine2: Infra route 10.0.72.67 present
in RIB
2019-10-01 16:11:47,106 INFO     ftriage:     main:622  Found peer-node a-leaf101 and IF:
Eth1/54 in candidate list
2019-10-01 16:12:09,836 INFO     ftriage:     main:839  L3 packet Seen on a-leaf101 Ingress:
Eth1/54 Egress: Eth1/30 (Po5) Vnid: 11470
2019-10-01 16:12:09,952 INFO     ftriage:   pktrec:490  a-leaf101: Collecting transient losses
snapshot for LC module: 1
2019-10-01 16:12:30,991 INFO     ftriage:     nxos:1404 a-leaf101: nxos matching rule id:4659
scope:84 filter:65534
```

```
2019-10-01 16:12:32,327 INFO     ftriage:     main:522  Computed egress encap string vlan-1075
2019-10-01 16:12:32,333 INFO     ftriage:     main:313  Building egress BD(s), Ctx
2019-10-01 16:12:34,559 INFO     ftriage:     main:331  Egress Ctx Prod:Vrf1
2019-10-01 16:12:34,560 INFO     ftriage:     main:332  Egress BD(s): Prod:Bd1
2019-10-01 16:12:37,704 INFO     ftriage:  unicast:1252 a-leaf101: Enter dbg_sub_nexthop with
Local inst: eg infra: False glbs.dipo: 10.0.72.67
2019-10-01 16:12:37,705 INFO     ftriage:  unicast:1257 a-leaf101: dbg_sub_nexthop invokes
dbg_sub_eg for ptep
2019-10-01 16:12:37,705 INFO     ftriage:  unicast:1784 a-leaf101: <- is egress node
2019-10-01 16:12:37,911 INFO     ftriage:  unicast:1833 a-leaf101: Dst EP is local
2019-10-01 16:12:37,912 INFO     ftriage:     misc:657  a-leaf101: EP if(Po5) same as egr
if(Po5)
2019-10-01 16:12:38,172 INFO     ftriage:     misc:657  a-leaf101: Dst IP is present in SUG L3
tbl
2019-10-01 16:12:38,564 INFO     ftriage:     misc:657  a-leaf101: RW seg_id:11470 in SUG same
as EP segid:11470
fTriage Status: {"dbgFtriage": {"attributes": {"operState": "Idle", "pid": "0", "apicId": "0",
"id": "0"}}}
fTriage Status: {"dbgFtriage": {"attributes": {"operState": "Idle", "pid": "0", "apicId": "0",
"id": "0"}}}
```

There is a large amount of data in the fTriage. Some of the most important fields are highlighted. Note that that the path of the packet was 'leaf205 (Pod 2) > spine3 (Pod 2) > spine2 (Pod 1) > leaf101 (Pod 1)'. All forwarding decisions and contract lookups made along the way are also visible.

Note that if this was a Layer 2 flow, the syntax of the fTriage would need to be set to something like:

```
ftriage bridge -ii LEAF:205 -dmac 00:00:11:11:22:22
```

## Proxied requests where the EP is not in COOP

Before considering specific failure scenarios, there is one more piece to discuss related to unicast forwarding over Multi-Pod. What happens if the destination endpoint is unknown, the request is proxied, and the endpoint is not in COOP?

In this scenario, the packet/frame is sent to the spine and a glean request is generated.

When the spine generates a glean request, the original packet is still preserved in the request however, the packet receives ethertype 0xfff2 which is a Custom Ethertype reserved for gleans. For this reason, it will not be easy to interpret these messages in packet capture tools such as Wireshark.

The outer Layer 3 destination is also set to 239.255.255.240 which is a reserved multicast group specifically for glean messages. These should be flooded across the fabric and any egress leaf switches that have the destination subnet of the glean request deployed will generate an ARP request to resolve the destination. These ARPs are sent from the BD Subnet IP Address configured (therefore proxy requests can't resolve the location of Silent/Unknown endpoints if Unicast Routing is disabled on a Bridge Domain).

The reception of the glean message on the egress leaf and the subsequently generated ARP and received ARP response can be verified through the following command:

## Glean ARP verification

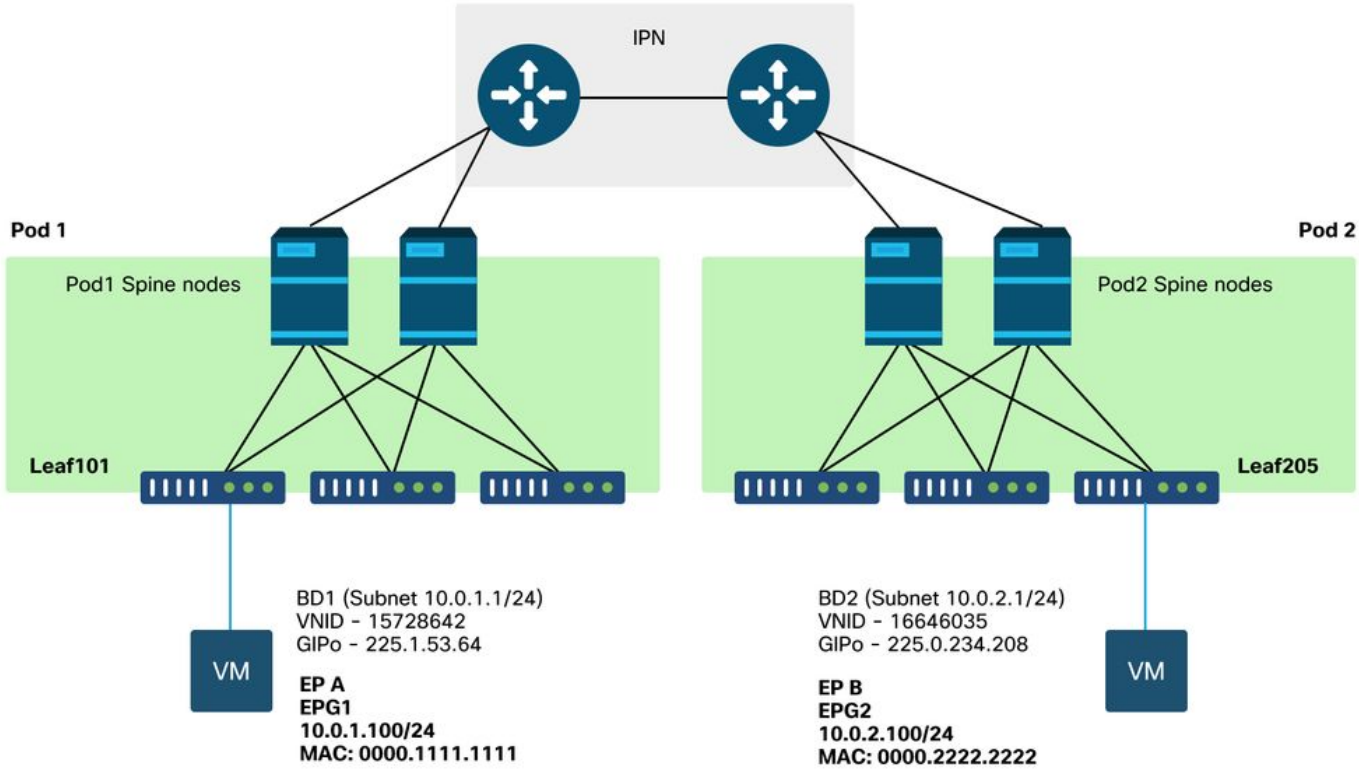
```
a-leaf205# show ip arp internal event-history event | grep -F -B 1 192.168.21.11
...
73) Event:E_DEBUG_DSF, length:127, at 316928 usecs after Wed May  1 08:31:53 2019
Updating epm ifidx: 1a01e000 vlan: 105 ip: 192.168.21.11, ifMode: 128 mac: 8c60.4f02.88fc   <<<
Endpoint is learned
75) Event:E_DEBUG_DSF, length:152, at 316420 usecs after Wed May  1 08:31:53 2019
log_collect_arp_pkt; sip = 192.168.21.11; dip = 192.168.21.254; interface = Vlan104;info = Garp
Check adj:(nil)  <<< Response received
77) Event:E_DEBUG_DSF, length:142, at 131918 usecs after Wed May  1 08:28:36 2019
log_collect_arp_pkt; dip = 192.168.21.11; interface = Vlan104;iod = 138; Info = Internal Request
Done   <<< ARP request is generated by leaf
78) Event:E_DEBUG_DSF, length:136, at 131757 usecs after Wed May  1 08:28:36 2019   <<< Glean
received, Dst IP is in BD subnet
log_collect_arp_glean;dip = 192.168.21.11;interface = Vlan104;info = Received pkt Fabric-Glean:
1
79) Event:E_DEBUG_DSF, length:174, at 131748 usecs after Wed May  1 08:28:36 2019
log_collect_arp_glean; dip = 192.168.21.11; interface = Vlan104; vrf = CiscoLive2019:vrf1; info
= Address in PSVI subnet or special VIP   <<< Glean Received, Dst IP is in BD subnet
```

For reference, glean messages being sent to 239.255.255.240 is why this group needs to be included in the Bidirectional PIM group range on the IPN.

## Multi-Pod Troubleshooting Scenario #1 (Unicast)

In the following topology, EP B cannot communicate with EP A.

### Troubleshooting topology




Note that many of the problems seen for Multi-Pod forwarding are identical to problems seen in a Single Pod. For this reason, problems specific to Multi-Pod are focused on.

While following the unicast troubleshooting workflow described earlier, note that the request is proxied but the spine nodes in Pod 2 do not have the destination IP in COOP.

**Cause: Endpoint Missing in COOP**

As discussed earlier, COOP entries for remote Pod endpoints are populated from BGP EVPN information. As a result, it is important to determine:

a.) Does the source Pod (Pod 2) spine have it in EVPN?

```
a-spine4# show bgp l2vpn evpn 10.0.1.100 vrf overlay-1
<no output>
```

b.) Does the remote Pod (Pod 1) spine have it in EVPN?

```
a-spine1# show bgp l2vpn evpn 10.0.1.100 vrf overlay-1
Route Distinguisher: 1:16777199    (L2VNI 1)
BGP routing table entry for [2]:[0]:[15728642]:[48]:[0050.5681.3ee6]:[32]:[10.0.1.100]/272,
version 11751 dest ptr 0xafbf8192
Paths: (1 available, best #1)
Flags: (0x00010a 00000000) on xmit-list, is not in rib/evpn
Multipath: eBGP iBGP

 Advertised path-id 1
 Path type: local 0x4000008c 0x0 ref 0 adv path ref 1, path is valid, is best path
 AS-Path: NONE, path locally originated
  0.0.0.0 (metric 0) from 0.0.0.0 (192.168.1.101)
    Origin IGP, MED not set, localpref 100, weight 32768
    Received label 15728642 2392068
    Extcommunity:
        RT:5:16

          Path-id 1 advertised to peers:
```

The Pod 1 spine has it and the next-hop IP is 0.0.0.0; this means it was exported from COOP locally. Note, however, that the 'Advertised to peers' section does not include the Pod 2 spine nodes.

c.) Is BGP EVPN up between Pods?

```
a-spine4# show bgp l2vpn evpn summ vrf overlay-1

Neighbor            V      AS   MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
192.168.1.101   4 65000       57380    66362            0     0      0  00:00:21  Active
192.168.1.102   4 65000       57568    66357            0     0      0  00:00:22  Active
```

Notice in the above output that the BGP EVPN peerings are down between Pods. Anything besides a numeric value in the State/PfxRcd column indicates that the adjacency is not up. Pod 1 EPs aren't learned through EVPN and aren't imported into COOP.

If this issue is seen verify the following:

1. Is OSPF up between the spine nodes and the connected IPNs?
2. Do the spine nodes have routes learned through OSPF for the remote spine IPs?
3. Does the full path across the IPN support jumbo MTU?
4. Are all protocol adjacencies stable?

**Other possible causes**

If the endpoint is not in the COOP database of any Pod and the destination device is a silent host (not learned on any leaf switch in the fabric), verify that the fabric glean process is working correctly. For this to work:

- Unicast Routing must be enabled on the BD.
- The destination must be in a BD subnet.
- The IPN must be providing multicast routing service for the 239.255.255.240 group.

The multicast portion is covered more in the next section.

# Multi-Pod broadcast, unknown unicast, and multicast (BUM) forwarding overview

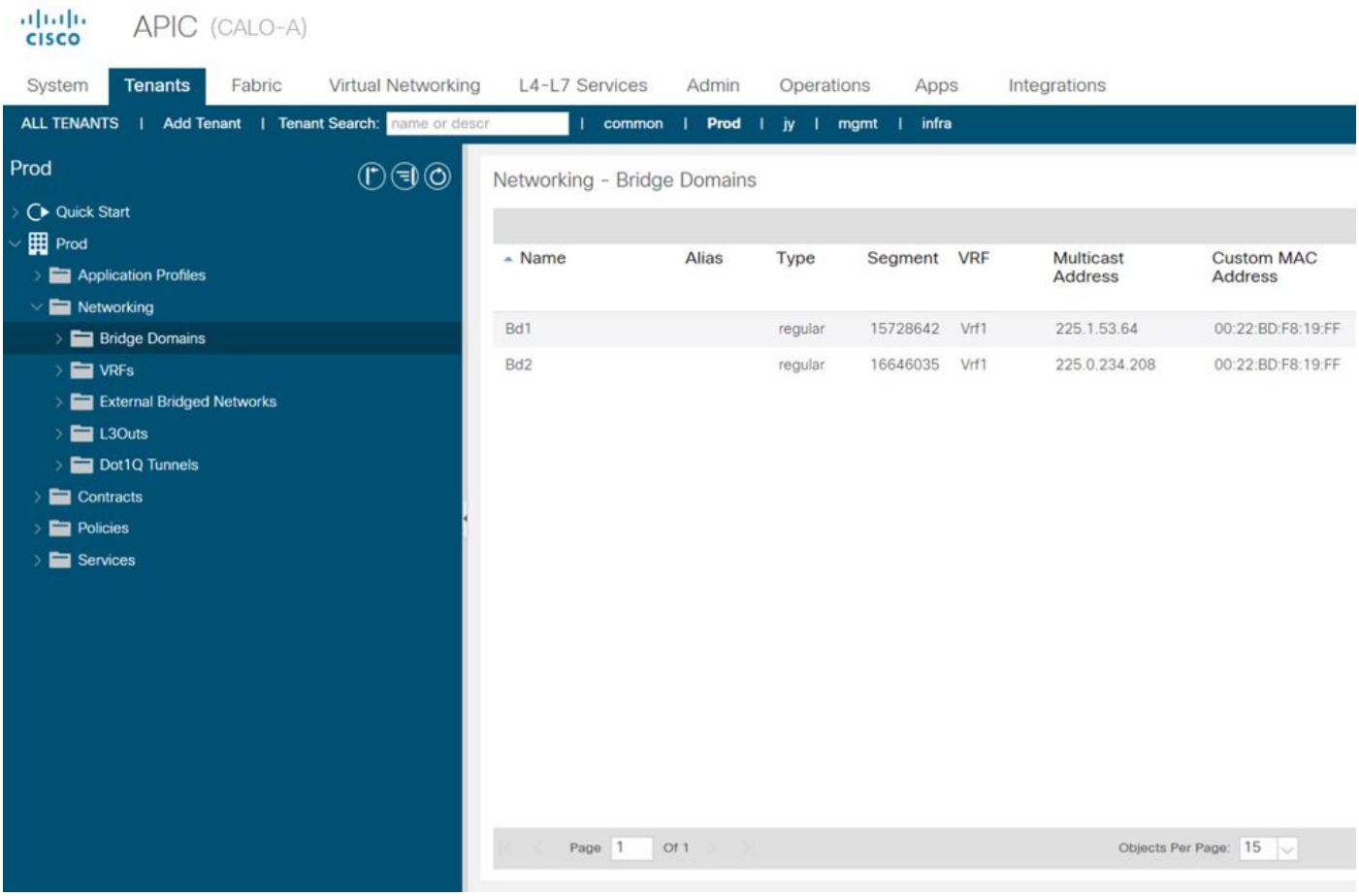
In ACI, traffic is flooded via overlay multicast groups in many different scenarios. For example, flooding occurs for:

- Multicast and broadcast traffic.
- Unknown unicast that must be flooded.
- Fabric ARP glean messages.
- EP announce messages.

Many features and functionality rely on BUM forwarding.

Within ACI, all Bridge Domains are allocated a multicast address known as a Group IP Outer (or GIPo) address. All traffic that must be flooded within a Bridge Domain is flooded on this GIPo.

## BD GIPo in GUI




The object can be queried directly on one of the APICs.

### BD GIPo in Moquery

```
a-apic1# moquery -c fvBD -f 'fv.BD.name=="Bd1"'
Total Objects shown: 1

# fv.BD
name                       : Bd1
OptimizeWanBandwidth       : no
annotation                 :
arpFlood                   : yes
bcastP                     : 225.1.53.64
childAction                :
configIssues               :
descr                      :
dn                         : uni/tn-Prod/BD-Bd1
epClear                    : no
epMoveDetectMode           :
extMngdBy                  :
hostBasedRouting           : no
intersiteBumTrafficAllow   : no
intersiteL2Stretch         : no
ipLearning                 : yes
ipv6McastAllow             : no
lcOwn                      : local
limitIpLearnToSubnets      : yes
llAddr                     : ::
mac                        : 00:22:BD:F8:19:FF
mcastAllow                 : no
modTs                      : 2019-09-30T20:12:01.339-04:00
monPolDn                   : uni/tn-common/monepg-default
```

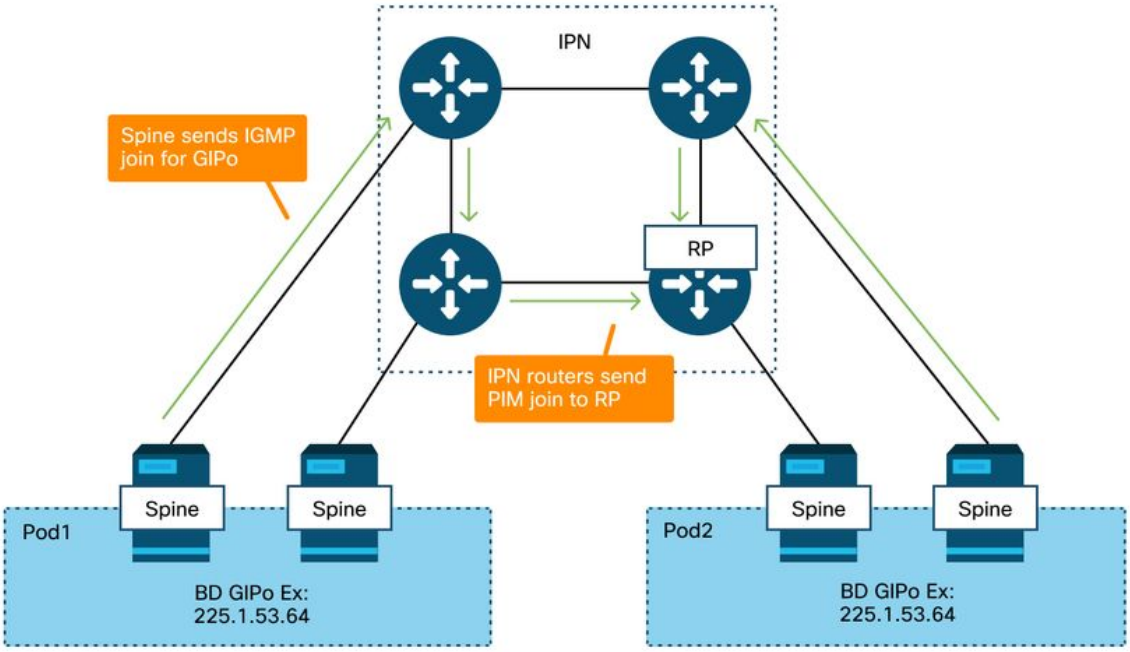
```
mtu                     : inherit
multiDstPktAct          : bd-flood
nameAlias               :
ownerKey                :
ownerTag                :
pcTag                   : 16387
rn                      : BD-Bd1
scope                   : 2392068
seg                     : 15728642
status                  :
type                    : regular
uid                     : 16011
unicastRoute            : yes
unkMacUcastAct          : proxy
unkMcastAct             : flood
v6unkMcastAct           : flood
vmac                    : not-applicable
```

The above information about GIPo flooding is true regardless whether Multi-Pod is used or not. The additional portion of this that pertains to Multi-Pod is the multicast routing on the IPN.

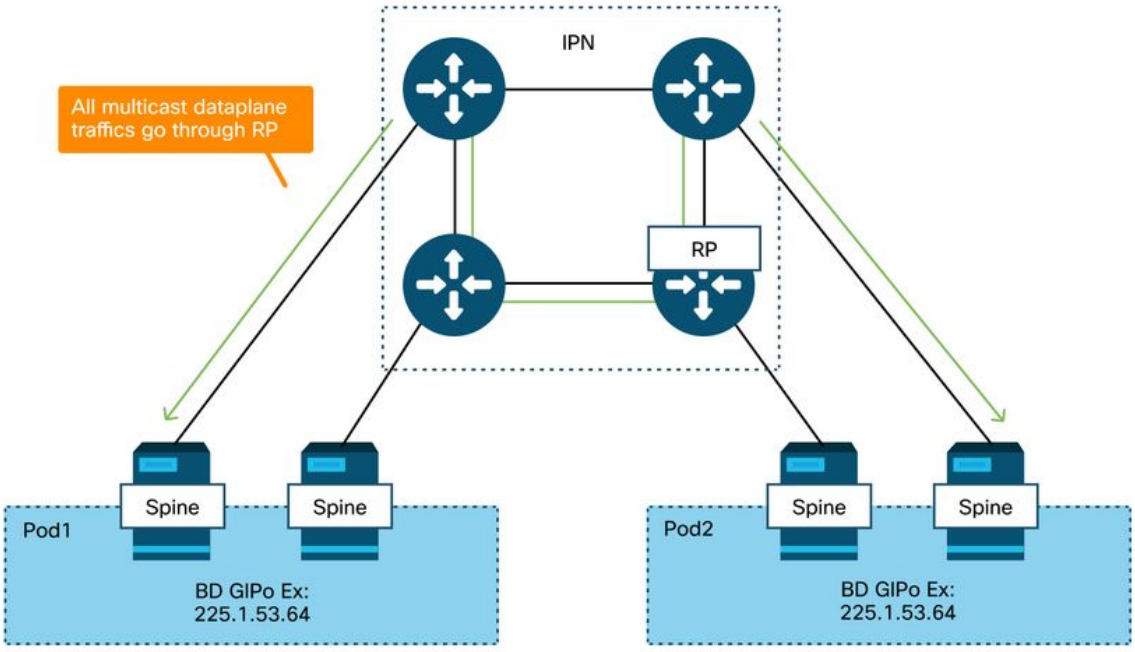
IPN Multicast Routing involves the following:

- Spine nodes act as multicast hosts (IGMP only). They do not run PIM.
- If a BD is deployed in a Pod, then one spine from that pod will send an IGMP join on one of its IPN-facing interfaces. This functionality is striped across all spine nodes and IPN-facing interface over many groups.
- The IPNs receive these joins and send PIM joins towards the Bidirectional PIM RP.
- Because PIM Bidir is used, there are no (S,G) trees. Only (*,G) trees are used in PIM Bidir.
- All dataplane traffic sent to the GIPo goes through the RP.

## IPN multicast control plane



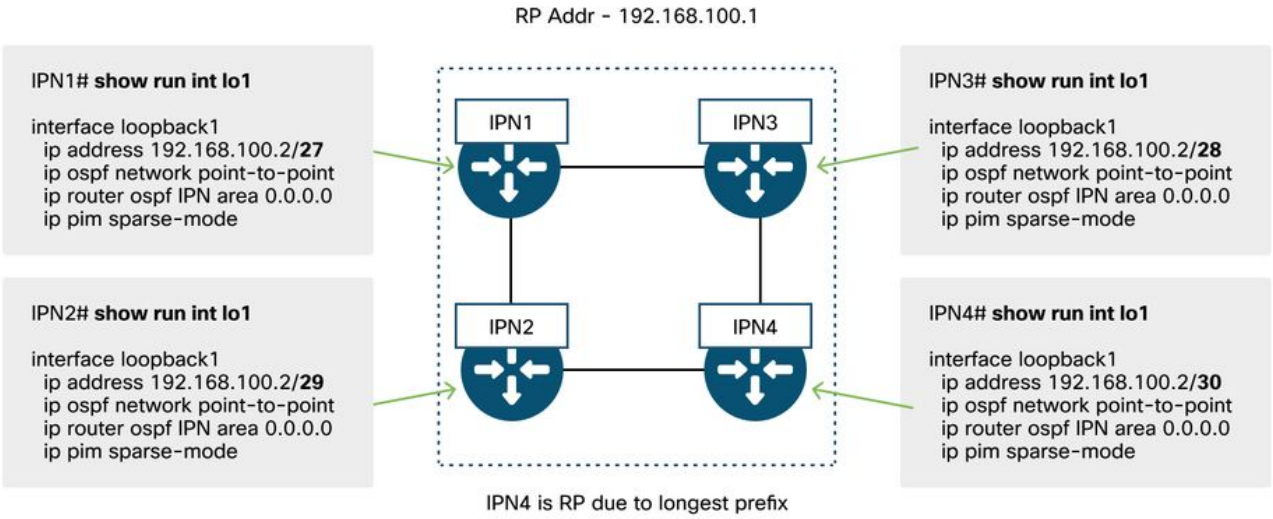
## IPN multicast dataplane




The only means of RP redundancy with PIM Bidir is to use Phantom. This is covered in detail within the Multi-Pod Discovery portion of this book. As a quick summary, note that with Phantom RP:

- All IPNs must be configured with the same RP address.
- The exact RP address must not exist on any device.
- Multiple devices advertise reachability to the subnet that contains the Phantom RP IP address. The advertised subnets should vary in subnet length so that all routers agree on who is advertising the best path for the RP. If this path is lost then convergence is dependent on the IGP.

## Phantom RP configuration




# Multi-Pod broadcast, unknown unicast, and multicast (BUM) troubleshooting workflow

## 1. First confirm if the flow is truly being treated as multi-destination by the fabric.

The flow will be flooded in the BD in these common examples:

- The frame is an ARP broadcast and ARP flooding is enabled on the BD.
- The frame is destined to a multicast group. Note that even if IGMP-snooping is enabled, the traffic is still always flooded into the fabric on the GIPo.
- The traffic is destined to a multicast group that ACI is providing multicast routing services for.
- The flow is a Layer 2 (bridged flow) and the destination MAC address is unknown and the unknown unicast behavior on the BD is set to 'Flood'.

The easiest way to determine which forwarding decision will be made is with an ELAM.

## 2. Identify the BD GIPo.

Refer to the section earlier in this chapter that talks about this. Spine ELAMs can also be run through the ELAM Assistant App to verify that the flooded traffic is being received.

## 3. Verify the multicast routing tables on the IPN for that GIPo.

The outputs to do this would vary depending on the IPN platform in use, but at a high level:

- All IPN routers must agree on the RP and the RPF for this GIPo must point to this tree.
- One IPN router connected to each Pod should be getting an IGMP join for the group.

## Multi-Pod Troubleshooting Scenario #2 (BUM Flow)

This scenario would cover any scenario that involves ARP not being resolved across Multi-Pod or BUM scenarios (unknown unicast, etc.).

There are several common possible causes here.

**Possible cause 1: Multiple routers own the PIM RP address**

With this scenario, the ingress leaf floods the traffic (verify with ELAM), the source Pod receives and floods the traffic, but the remote Pod does not get it. For some BDs, flooding works, but for others it doesn't.

On the IPN, run 'show ip mroute <GIPo address>' for the GIPo to see that the RPF tree points to multiple, different routers.

If this is the case check the following:

- Verify that the actual PIM RP address isn't configured anywhere. Any device that owns that actual RP address would see a local /32 route for it.
- Verify that multiple IPN routers aren't advertising the same prefix length for the RP in the Phantom RP scenario.

**Possible cause 2: IPN routers aren't learning routes for the RP Address**

In the same way as the first possible cause, here the flooded traffic is failing to leave the IPN. The output of 'show ip route <rp address>' on each IPN router would show the locally configured prefix-length only rather than what the other routers are advertising.

The result of this is that each device thinks it is the RP even though the real RP IP address isn't configured anywhere.

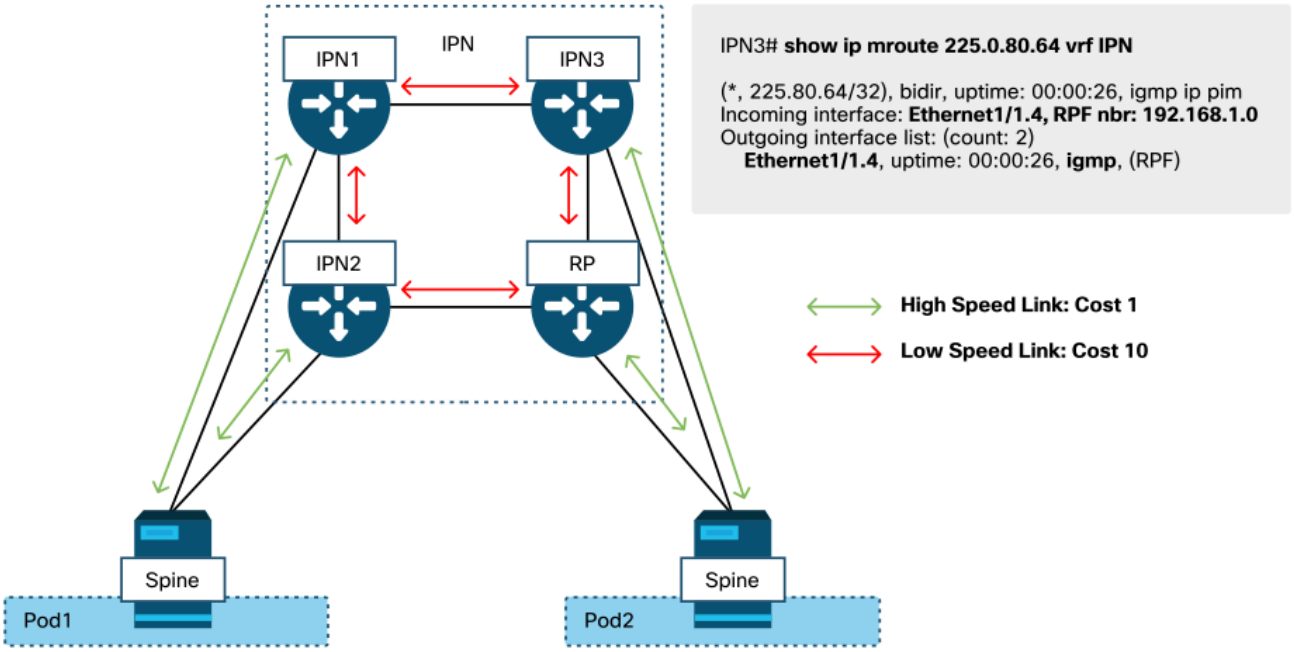
If this is the case. check the following:

- Verify that routing adjacencies are up between IPN routers. Verify that the route is in the actual protocol database (such as the OSPF database).
- Verify that all loopbacks that are supposed to be candidate RP's are configured as OSPF point-to-point network types. If this network type is not configured then each router will always advertise a /32 prefix-length regardless of what is actually configured.

**Possible cause 3: IPN routers aren't installing the GIPo route or the RPF points to ACI**

As mentioned earlier, ACI does not run PIM on its IPN-facing links. This means that the IPN's best path towards the RP should never point to ACI. The scenario where this could happen would be if multiple IPN routers are connected to the same spine and a better OSPF metric is seen through the spine than directly between IPN routers.

**RPF interface toward ACI**




To resolve this issue:

- Ensure that routing protocol adjacencies between IPN routers are up.
- Increase the OSPF cost metrics for the IPN-facing links on the spine nodes to a value that will make that metric less preferable than the IPN-to-IPN links.

## Other references

Prior to ACI software 4.0, some challenges were experienced regarding the usage of COS 6 by external devices. Most of these issues have been addressed through 4.0 enhancements but for more info, please refer to CiscoLive session "BRKACI-2934 - Troubleshooting Multi-Pod" and the "Quality of Service" section.