Cisco Dynamic Fabric Automation Production Troubleshooting Guide

First Published: August 18, 2016

Conventions (all documentation)

This document uses the following conventions.

Convention	Indication
bold font	Commands and keywords and user-entered text appear in **bold** font.
italic font	Document titles, new or emphasized terms, and arguments for which you supply values are in *italic* font.
[ ]	Elements in square brackets are optional.
{x | y | z }	Required alternative keywords are grouped in braces and separated by vertical bars.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
courier font	Terminal sessions and information the system displays appear in courier font.
< >	Nonprinting characters such as passwords are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

Note: Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the manual.

Caution: **Means *reader be careful.* In this situation, you might perform an action that could result in equipment damage or loss of data.**

Warning: **IMPORTANT SAFETY INSTRUCTIONS**

Means danger. You are in a situation that could cause bodily injury. Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents. Use the statement number provided at the end of each warning to locate its translation in the translated safety warnings that accompanied this device.

SAVE THESE INSTRUCTIONS

Regulatory: Provided for additional information and to comply with regulatory and customer requirements.

Legal Information

Cisco Trademark

Cisco Copyright

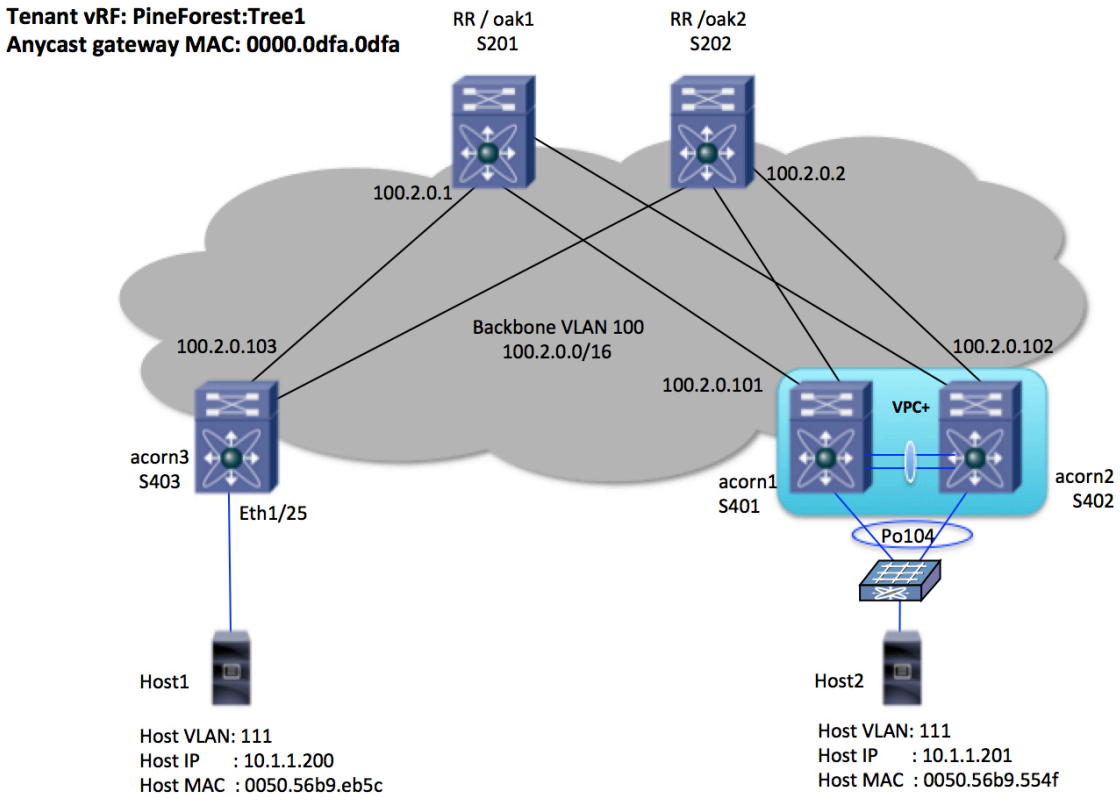
Contents

1 Introduction

This document is intended to focus on troubleshooting procedures or methodology post Cisco DCNM POAP programming for Dynamic Fabric Automation (DFA) environments (which is covered in detail at Cisco Dynamic Fabric Automation Troubleshooting Guide).

With the use of this document, network administrators are provided with the information on how to resolve or isolate issues when troubleshooting is required in a DFA Fabric with a fabricpath underlay.This document is not intended to be a best practice or deployment guide. For information on configuring and deploying DFA networks, see the Related Links section of this document. The Cisco Nexus 5000/6000 Series switches will be used as the leaf and border leaf switches in this topology, but we will point out where some differences may be with the Cisco Nexus 7700 Series switches. Cisco Nexus 7000 series switches are used as the spine switches in this topology. The following are details of the hardware and software used for this document:

- Cisco DCNM 7.2 – For provisioning/configuring the fabric underlay.
- Leaf and Border Leaf switches: Cisco Nexus 6001 and 6004
    - 7.1(2)N1(1)
- Spine / Route-Reflectors: Cisco Nexus 7010
    - 6.2(12)
- Topology: The following reference topology will be used in the below troubleshooting examples:



2 Profile Instantiation

2.1 Understanding Profile Instantiation Process

Auto-configuration is the instantiation of a partition/network by applying parameter instances to partition/network profiles (i.e. templates). A DFA partition is equivalent to a VRF, whereas a DFA network within a DFA partition is equivalent to a VLAN/Segment ID in this particular VRF.

Profiles and parameter instances are downloaded by the leaf switches from the LDAP database.

The LDAP database can be residing in the Cisco DCNM or can be external to the Cisco DCNM. In this document, focus will be on the LDAP database included in the Cisco DCNM.

Auto-configuration is triggered by Host Mobility Manager (HMM) by either data traffic in a particular VLAN, or control plane (VDP protocol signaling between N1Kv and leaf signaling the ask for a particular VNI). For the HMM troubleshooting specifics refer to the Host Mobility via ARP and VDP section.

This section assumes HMM was triggered successfully and the switch is in the following order.

1. In the process of obtaining the matching profile and parameter instance from the external LDAP database for the particular VLAN/Segment ID (responsibility of Asset Database Manager process [**ADBM**])

2. Once the above is completed, the matching profile will be instantiated on the leaf with the parameter instance obtained (responsibility of Port Profile Manager process [**PPM**])

2.2 Debugging profile instantiation problems

2.2.1 Verify ADBM is able to talk to the LDAP database

If auto-configuration for a partition/network on a leaf switch(es) is unsuccessful, a syslog explaining why should be displayed. An example is shown below:

```
HMM-3-AUTO_CONF_PROFILE_ERROR  hmm VDP Host: Remote Database error:

HMM-3-AUTO_CONF_PROFILE_ERROR  hmm Data snooping Host: Remote Database error:
```

This typically happens when ADBM was unable to find an entry in or reach the LDAP database. Start by looking at the communication between the ADBM process on the switch and LDAP on Cisco DCNM (or external server). The following tables are from the perspective of the ADBM process.

```
acorn1%PineForest:Tree1# show fabric database statistics
DB-Type         Requests    Dispatched  Not dispatched  Re-dispatched
-------------------------------------------------------------------
network                0             0               0              0
cabling                0             0               0              0
profile                0             0               0              0
partition              0             0               0              0
bl-dci                 0             0               0              0
-------------------------------------------------------------------
TOTAL                  0             0               0              0

Per Database stats:
 T Prot Server/DB                       Reqs     OK  NoRes    Err  TmOut   Pend
-------------------------------------------------------------------------------
*ne ldap dfa-dcnm1                         0      0      0      0      0      0
*pr ldap dfa-dcnm1                         0      0      0      0      0      0
*pa ldap dfa-dcnm1                         0      0      0      0      0      0
Legend:
  T-Type (ne-network, ca-cabling, pr-profile, pa-partition, bl-bl-dci)
  *-Active Server
```

The *Per Database stats* table displays the requests that went out to the active server *dfa-dcnm1*.

Section 2.2.2 below will provide examples as to scenarios in which you may see these counters increase.

2.2.2 Manual re-verification via "test fabric database" commands

Manual re-verification can be accomplished via CLI to test the communication between ADBM process on the switch and the LDAP server (LDAP server is running within Cisco DCNM in the examples within this White Paper, but can be external to Cisco DCNM if needed).

Note -This is just to test database querying and will not trigger profile instantiation.

```
acorn1%PineForest:Tree1# test fabric database client-req type network vlan 112
acorn1%PineForest:Tree1# show fabric database statistics
DB-Type         Requests    Dispatched  Not dispatched  Re-dispatched
-----------------------------------------------------------------
network                1             0               1              0
cabling                0             0               0              0
profile                0             0               0              0
partition              0             0               0              0
bl-dci                 0             0               0              0
-----------------------------------------------------------------
TOTAL                  1             0               1              0

Per Database stats:
 T Prot Server/DB                       Reqs     OK  NoRes    Err  TmOut   Pend
-------------------------------------------------------------------------------
*ne ldap dfa-dcnm1                        0      0      0      0      0      0
*pr ldap dfa-dcnm1                        0      0      0      0      0      0
*pa ldap dfa-dcnm1                        0      0      0      0      0      0
Legend:
  T-Type (ne-network, ca-cabling, pr-profile, pa-partition, bl-bl-dci)
  *-Active Server
```

The above test command manually triggered 1 network request (which as mentioned previously could also be triggered via HMM), but the request was not dispatched to the LDAP server. As a result, no requests went out via LDAP protocol to query the database of the active server *dfa-dcnm1*.

The reason no LDAP request was sent was due to an incomplete/ambiguous request (no Mobility Domain specified).

Below is an example of a working scenario of a CLI triggered network request for VLAN tag 112 in mobility domain PineForest on a leaf switch acorn1.

```
acorn1%PineForest:Tree1# test fabric database client-req type network vlan 112 mdomain PineForest
acorn1%PineForest:Tree1# show fabric database statistics
DB-Type         Requests    Dispatched  Not dispatched  Re-dispatched
-----------------------------------------------------------------
network                2             1               1              0
cabling                0             0               0              0
profile                0             0               0              0
partition              0             0               0              0
bl-dci                 0             0               0              0
-----------------------------------------------------------------
TOTAL                  2             1               1              0

Per Database stats:
 T Prot Server/DB                       Reqs     OK  NoRes    Err  TmOut   Pend
-------------------------------------------------------------------------------
*ne ldap dfa-dcnm1                        1      1      0      0      0      0
*pr ldap dfa-dcnm1                        0      0      0      0      0      0
*pa ldap dfa-dcnm1                        0      0      0      0      0      0
Legend:
  T-Type (ne-network, ca-cabling, pr-profile, pa-partition, bl-bl-dci)
  *-Active Server
```

One network request was dispatched to the LDAP server, and the LDAP server sent a request for a network to the active server and got an OK response (so a hit was found in the database).

Verify this on Cisco DCNM:

We can see below that there is no error in querying the database. One matching network entry is found for vlan-id 112 and mobility domain (MD) pineforest.

```
ssh root@dfa-dcnm1
tail -f /var/log/ldap.log
Nov 10 06:42:00 dfa-dcnm1 slapd[21215]: conn=4435 op=33 SRCH base="ou=networks,dc=cisco,dc=com" scope=2
deref=0 filter="(&(vlanId=112)(mobilityDomainId=pineforest))"
Nov 10 06:42:00 dfa-dcnm1 slapd[21215]: conn=4435 op=33 SEARCH RESULT tag=101 err=0 nentries=1 text=
```

There are a few possible errors that can occur when querying the database.

1. No entry in the database.
2. Security misconfiguration.

Below are output examples of each:

1. No entry in the database:

```
acorn1%PineForest:Tree1# test fabric database client-req type network vlan 152 if port-channel104 mdo-
main PineForest
acorn1%PineForest:Tree1# show fabric database statistics
DB-Type         Requests    Dispatched  Not dispatched  Re-dispatched
-------------------------------------------------------------------
network                3             2               1              0
cabling                0             0               0              0
profile                0             0               0              0
partition              0             0               0              0
bl-dci                 0             0               0              0
-------------------------------------------------------------------
TOTAL                  3             2               1              0

Per Database stats:
 T Prot Server/DB                       Reqs     OK  NoRes    Err  TmOut   Pend
-------------------------------------------------------------------------------
*ne ldap dfa-dcnm1                         2      1      1      0      0      0
*pr ldap dfa-dcnm1                         0      0      0      0      0      0
*pa ldap dfa-dcnm1                         0      0      0      0      0      0
Legend:
  T-Type (ne-network, ca-cabling, pr-profile, pa-partition, bl-bl-dci)
  *-Active Server
```

The request went out, but there was no response from the active server (the NoRes counter increased after the request). What does Cisco DCNM show?

```
ssh root@dfa-dcnm1
tail -f /var/log/ldap.log
Nov 10 07:16:33 dfa-dcnm1 slapd[21215]: conn=4435 op=35 SRCH base="ou=networks,dc=cisco,dc=com" scope=2
deref=0 filter="(&(vlanId=152)(mobilityDomainId=pineforest))"
Nov 10 07:16:33 dfa-dcnm1 slapd[21215]: conn=4435 op=35 SEARCH RESULT tag=101 err=0 nentries=0 text=
```

No error, no matching entry found in the database, LDAP has nothing to return.

Solution: Create matching network in database.

2. Security misconfiguration

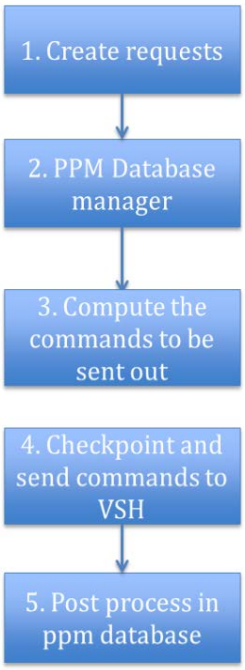
```
acorn1%PineForest:Tree1# test fabric database client-req type network vlan 152 if port-
channel104 mdomain PineForest
acorn1%PineForest:Tree1# show fabric database statistics
DB-Type         Requests    Dispatched  Not dispatched  Re-dispatched
-------------------------------------------------------------------
network                3             2               1              0
cabling                0             0               0              0
profile                0             0               0              0
partition              0             0               0              0
bl-dci                 0             0               0              0
-------------------------------------------------------------------
TOTAL                  3             2               1              0

Per Database stats:
 T Prot Server/DB                      Reqs     OK  NoRes    Err  TmOut   Pend
-------------------------------------------------------------------------------
*ne ldap dfa-dcnm1                        3      1      1      1      0      0
*pr ldap dfa-dcnm1                        0      0      0      0      0      0
*pa ldap dfa-dcnm1                        0      0      0      0      0      0
Legend:
  T-Type (ne-network, ca-cabling, pr-profile, pa-partition, bl-bl-dci)
  *-Active Server
```

Request went out with error: As can be seen above with the Err column increasing by 1, there was likely a database security misconfiguration, such as incorrect user/pw on Cisco DCNM or Leaf switch ***db-security user*** mis-configuration.

Solution: Ensure the password for the LDAP user matches between switch and Cisco DCNM.

2.3 ADBM can talk to the LDAP server, but the profile is still not successfully applied

Below is a workflow example of what occurs after the profile and its respective parameters are provided by the LDAP server, and instantiation takes place.



2.3.1 Do a manual pull of the profile/instance and verify it can get applied

Below is a working case of a manual (i.e. not triggered by traffic or VDP, but by the user through CLI) pull that instantiates vlan 112 on port-channel interface 104 (an interface part of mobility domain PineForest):

```
acorn1%PineForest:Tree1# fabric database auto-pull dot1q 112 interface port-channel 104
acorn1%PineForest:Tree1# show fabric database host statistics
ADBM Requests                  2
ADBM Responses                 2
Profile Apply Received         1
Profile vPC Queued             0
Profile Local Apply Queued     0
Profile Local UnApply Queued   0
Profile Apply Sent             1
Profile Apply Responses        5
Profile Apply Success          1
Profile Commands               4
Profile High Queue adds        1
Outstanding vlan requests      0
Outstanding adbm requests      0
Outstanding Profile Applies    0
Outstanding vPC Profile Applies 0

acorn1%PineForest:Tree1# show fabric database host dot1q 112
instance_index 0
Got Local originated vlan type trigger at 16:48:52
Number of associated interfaces: 1
Sent to Database Manager at 16:48:53
Received Parameters from Database Manager at 16:48:53
Sent to vPC peer at 16:48:54
Completed executing all commands on vPC peer at 16:48:54
Displaying Data Snooping Ports
Interface      Encap      Flags State
```

```
Po104          112        LX    Profile Active

acorn1%PineForest:Tree1# show fabric database host dot1q 112 internal
instance_index 0
Got Local originated vlan type trigger at 16:48:52
Number of associated interfaces: 1
Sent to Database Manager at 16:48:53
Received Parameters from Database Manager at 16:48:53
Sent to vPC peer at 16:48:54
Completed executing all commands on vPC peer at 16:48:54
Parent Data:conf flag 0x0 vni 31002 vlan 0 l3_vni 0 bd 0 vrf PineForest:Tree1 mode 0 mcec 104
flags0x1000110
xlated_vlan 1604 vni 31002 original_vlan 112 mob_dom PineForest
Displaying Data Snooping Ports
Interface      Encap      Flags State
Po104          112        LX    Profile Active
conf flag 0x0 vni 0 vlan 0 l3_vni 0 bd 0 vrf  mode 0 mcec 104 flags0x1000010
xlated_vlan 1604 vni 31002 original_vlan 112 mob_dom PineForest

acorn1%PineForest:Tree1# show fabric database host vni 31002 internal
instance_index 59
Got Local originated vlan type trigger at 16:48:53
Number of associated interfaces: 1
The profile will be checked for aging in 850 seconds
Got static profile
Displaying parameters for profile defaultNetworkUniversalTfProfile and instance instance_vni_31002_59
parameter 0: $gatewayIpAddress=10.1.2.1
parameter 1: $netMaskLength=24
parameter 2: $vlanId=1604
parameter 3: $segmentId=31002
parameter 4: $vrfName=PineForest:Tree1
parameter 5: $gatewayIpAddress=10.1.2.1
parameter 6: $netMaskLength=24
parameter 7: $dhcpServerAddr=5.0.3.91
parameter 8: $vrfDhcp=management
parameter 9: $gatewayIpv6Address=
parameter 10: $prefixLength=
parameter 11: $mtuValue=
parameter 12: $include_vrfSegmentId=50010
parameter 13: $vlanId=1604
parameter 14: $asn=60100
Sent Apply to Configuration Manager at 16:48:53
Completed executing all commands at 16:48:54
Sent to vPC peer at 16:48:54
Completed executing all commands on vPC peer at 16:48:55
Parent Data:conf flag 0x0 vni 31002 vlan 1604 l3_vni 0 bd 1604 vrf PineForest:Tree1 mode 2 mcec 0
flags0x1000110
xlated_vlan 1604 vni 31002 original_vlan 112 mob_dom PineForest
Displaying Data Snooping Ports
Interface      Encap      Flags State
Po104          1604       LX    Profile Active
conf flag 0x0 vni 0 vlan 1604 l3_vni 0 bd 1604 vrf  mode 0 mcec 104 flags0x1000010
xlated_vlan 1604 vni 31002 original_vlan 112 mob_dom PineForest
acorn1%PineForest:Tree1# sh runn | i apply.*31002
apply profile defaultNetworkUniversalTfProfile include-profile vrf-common-universal param-instance in-
stance_vni_31002_59 include-instance PineForest:Tree1_53
```

Looking at the configuration profile application history we see that the profile for PineForest:Tree1 was instantiated with VNI 31002 dynamically at 11/10/15 16:48:54.

```
acorn1%PineForest:Tree1# show system internal config-profile history apply success
================================================================================
Inst/Param List Name   : instance_vni_31002_59/param_list_name_instance_vni_31002_59
Include List/Inst Name : include_param_list_name_PineForest:Tree1_53/PineForest:Tree1_53
Refresh Profile Name   :
Refresh List/Inst Name : /
Operation Type         : Apply
Time Requested         : 11/10/15 16:48:53.430922
Time Responded         : 11/10/15 16:48:54.150491
Rollback Status        : Rollback not done
Configuration Type     : Auto Config
================================================================================
```

Some time later, it looks like the profile has been re-instantiated multiple times. Take note that the translate-to vlan id has changed (previously vni_31002_59, and below to vni_31002_90).

```
acorn1%PineForest:Tree1# fabric database auto-pull dot1q 112 interface port-channel 104
acorn1%PineForest:Tree1# show system internal config-profile history apply success
…
================================================================================
Inst/Param List Name   : instance_vni_31002_90/param_list_name_instance_vni_31002_90
Include List/Inst Name : include_param_list_name_PineForest:Tree1_53/PineForest:Tree1_53
Refresh Profile Name   :
Refresh List/Inst Name : /
Operation Type         : Apply
Time Requested         : 11/12/15 12:14:52.618863
Time Responded         : 11/12/15 12:14:53.282910
Rollback Status        : Rollback not done
Configuration Type     : Auto Config
================================================================================
```

Looking at the accounting log of the profile instantiation, you can see the actual commands being added to the switch at the “Time requested/responded” timestamps from the above config-profile history.

```
acorn1# show accounting log | i 12:14
Thu Nov 12 12:14:52 2015:type=update:id=5.0.1.111@pts/1:user=admin:cmd=fabric database auto-pull dot1q
112 interface port-channel104 (SUCCESS)
Thu Nov 12 12:14:52 2015:type=start:id=vsh.1110:user=root:cmd=
Thu Nov 12 12:14:52 2015:type=stop:id=vsh.1110:user=root:cmd=
Thu Nov 12 12:14:52 2015:type=update:id=ppm.3691:user=admin:cmd=__ppm_user admin ppm.3691 (SUCCESS)
Thu Nov 12 12:14:52 2015:type=update:id=ppm.3691:user=admin:cmd=terminal ask-on-term
/dev/shm/ppm_defaultNetworkUniversalTfProfile19795853590997_output.txt (SUCCESS)
Thu Nov 12 12:14:52 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; vlan 1635 (SUC-
CESS)
Thu Nov 12 12:14:52 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; vlan 1635 (SUC-
CESS)
Thu Nov 12 12:14:53 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; vlan 1635 ; vn-
segment 31002 (SUCCESS)
Thu Nov 12 12:14:53 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; vlan 1635 ; mode
fabricpath (SUCCESS)
Thu Nov 12 12:14:53 2015:type=update:id=ppm.3691:user=admin:cmd= (SUCCESS)
Thu Nov 12 12:14:53 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; interface Vlan1635
(SUCCESS)
Thu Nov 12 12:14:53 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; interface Vlan1635
; vrf member PineForest:Tree1 (SUCCESS)
Thu Nov 12 12:14:53 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; interface Vlan1635
; ip address 10.1.2.1/24 tag 12345 (SUCCESS)
```

```
Thu Nov 12 12:14:53 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; interface Vlan1635
; ip dhcp relay address 5.0.3.91 use-vrf management (SUCCESS)
Thu Nov 12 12:14:53 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; interface Vlan1635
; fabric forwarding mode anycast-gateway (SUCCESS)
Thu Nov 12 12:14:53 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; interface Vlan1635
; no shutdown (SUCCESS)
```

Two categories of possible errors/reasons for config-profile not being applied are:

1. **Pre-execution check failure** – Attempting to create a merged command tree using the existing and the new configurations being applied.

2. **Execution failure** – VSH attempting to push a particular command failed.

Below are output examples of each:

2.3.2 Pre-execution check failure

On the leaf switch of interest check the following:

```
acorn1%PineForest:Tree1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
acorn1%PineForest:Tree1(config)# apply profile defaultNetworkUniversalEfProfile include-profile vrf-
common-universal-bl param-instance instance_vni_31001_40 include-instance PineForest:Tree1_53
ERROR: Profile conflicts with another profile
acorn1%PineForest:Tree1(config)#
```

You can also check historical config-profile information if looking at an issue after the fact, as well as the accounting log to see if the commands were marked as Success or Failure.

Note: pre-execution check failures are not listed under the failed config-profile applications.

```
acorn1%PineForest:Tree1# show system internal config-profile history | last 13
================================================================================
Inst/Param List Name   : instance_vni_31001_40/
Include List/Inst Name : /PineForest:Tree1_53
Refresh Profile Name   :
Refresh List/Inst Name : /
Operation Type         : Apply
Time Requested         : 11/13/15 10:13:11.761935
Time Responded         : 11/13/15 10:13:11.762246
Failure Cmd           : Pre-execution check failure
Failure Reason         : Profile conflicts with another profile
Rollback Status        : Rollback not done
Configuration Type     : Manual Config
================================================================================
```

```
acorn1%PineForest:Tree1# show accounting log | i 10:13:1
Fri Nov 13 10:13:11 2015:type=update:id=5.0.1.111@pts/1:user=admin:cmd=configure terminal ; apply pro-
file defaultNetworkUniversalEfProfile include-profile vrf-com
mon-universal-bl param-instance instance_vni_31001_40 include-instance PineForest:Tree1_53 (FAILURE)
acorn1#
```

Look at what other profiles get instantiated with the same parameter set instance:

```
acorn1# show runn | i ^apply.*instance_vni_31001_40
apply profile defaultNetworkUniversalEfProfile include-profile vrf-common-universal param-instance in-
stance_vni_31001_40 include-instance PineForest:Tree1_53
```

Solution: Fix the include-profile conflict between the templates *vrf-common-universal-bl* and *vrf-common-universal*.

2.3.3 Execution failure

On the leaf switch of interest, check the following:

```
acorn1%PineForest:Tree1# conf t
acorn1%PineForest:Tree1(config)# apply profile vrf-tenant-profile param-instance param_inst_50010b
Message reported by command ::     config terminal
ERROR: Add segment id node failed as the same id is used by vlan 1502
Message reported by command ::     end
ERROR : cli_process_vlan_config_exit(295), command vn-segment 50010 FAILED
2015 Nov 12 12:42:55 acorn1 %ETHPORT-3-IF_ERROR_VLANS_ERROR: VLANs 1503 on Interface port-channel1 are
in error state. (Reason: Vlan is not allowed on Peer-link)
ERROR: Failed to write VSH commands
acorn1%PineForest:Tree1# 2015 Nov 12 12:43:02 acorn1 last message repeated 1 time
2015 Nov 12 12:43:02 acorn1 %VSHD-5-VSHD_SYSLOG_CONFIG_I: Configured from vty by admin on
5.0.1.111@pts/1
```

In the below output, you can see that “config terminal” was unsuccessfully completed, indicating one could not execute the commands to instantiate the downloaded profile/parameters on the switch.

```
acorn1%PineForest:Tree1# show system internal config-profile history apply fail | last 11
================================================================================
Inst/Param List Name   : param_inst_50010b/
Refresh List/Inst Name : /
Operation Type         : Apply
Time Requested         : 11/12/15 12:42:55.254258
Time Responded         : 11/12/15 12:42:56.562875
Failure Cmd            : config terminal
Failure Reason         :
Rollback Status        : Success
Configuration Type     : Manual Config
================================================================================
acorn1%PineForest:Tree1# show accounting log | i 12:4
Thu Nov 12 12:42:55 2015:type=start:id=vsh.3292:user=root:cmd=
Thu Nov 12 12:42:55 2015:type=stop:id=vsh.3292:user=root:cmd=
Thu Nov 12 12:42:55 2015:type=update:id=ppm.3691:user=admin:cmd=__ppm_user admin ppm.3691 (SUCCESS)
Thu Nov 12 12:42:55 2015:type=update:id=ppm.3691:user=admin:cmd=terminal ask-on-term /dev/pts/1 (SUC-
CESS)
Thu Nov 12 12:42:55 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; vlan 1503 (SUC-
CESS)
Thu Nov 12 12:42:55 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; vlan 1503 (SUC-
CESS)
Thu Nov 12 12:42:55 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; vlan 1503 ; vn-
segment 50010 (FAILURE)
Thu Nov 12 12:42:55 2015:type=update:id=ppm.3691:user=admin:cmd=configure terminal ; vlan 1503 ; mode
fabricpath (SUCCESS)
Thu Nov 12 12:42:55 2015:type=start:id=vsh.3299:user=root:cmd=
Thu Nov 12 12:42:55 2015:type=update:id=ppm.3691:user=admin:cmd= (SUCCESS)
Thu Nov 12 12:42:55 2015:type=stop:id=vsh.3299:user=root:cmd=
Thu Nov 12 12:42:56 2015:type=start:id=vsh.3306:user=root:cmd=
```

```
Thu Nov 12 12:42:56 2015:type=stop:id=vsh.3306:user=root:cmd=
Thu Nov 12 12:42:56 2015:type=start:id=vsh.3316:user=admin:cmd=
Thu Nov 12 12:42:56 2015:type=update:id=vsh.3316:user=admin:cmd=configure terminal ; no vlan 1503 (SUC-
CESS)
Thu Nov 12 12:42:56 2015:type=stop:id=vsh.3316:user=admin:cmd=
Thu Nov 12 12:42:56 2015:type=update:id=5.0.1.111@pts/1:user=admin:cmd=configure terminal ; apply pro-
file vrf-tenant-profile param-instance param_inst_50010b (FAILURE)


acorn1%PineForest:Tree1# show config-profile name vrf-tenant-profile

config-profile vrf-tenant-profile
  vlan $vrfVlanId
   vn-segment $vrfSegmentId
   mode fabricpath
  interface vlan $vrfVlanId
   vrf member $vrfName
   ip forward
   no ip redirects
   ipv6 forward
   no ipv6 redirects
   mtu 9192
   no shutdown
 applied: <param_inst_50010>
 applied: <param_inst_50011>

Look for the instances matching the applied ones and look for conflicts with the one we tried to apply.

acorn1%PineForest:Tree1# show param-list show-instance
....
Param List Name : param_list
        Name : bridgeDomainId   Type : integer
        Name : vrfBdName        Type : string
        Name : vrfName  Type : string
        Name : vrfSegmentId     Type : integer
        Name : vrfVlanId        Type : integer
        Param Instance Name : param_inst_50010
        Name : bridgeDomainId   Value : 0
        Name : vrfBdName        Value : __bd50010
        Name : vrfName  Value : PineForest:Tree1
        Name : vrfSegmentId     Value : 50010
        Name : vrfVlanId        Value : 1502
        Param Instance Name : param_inst_50010b
        Name : bridgeDomainId   Value : 0
        Name : vrfBdName        Value : __bd50010b
        Name : vrfName  Value : PineForest:Tree1
        Name : vrfSegmentId     Value : 50010
        Name : vrfVlanId        Value : 1503
        Param Instance Name : param_inst_50011
        Name : bridgeDomainId   Value : 0
        Name : vrfBdName        Value : __bd50011
        Name : vrfName  Value : OakForest:OakTree11
        Name : vrfSegmentId     Value : 50011
        Name : vrfVlanId        Value : 1501
....
```

One sees that " param_inst_50010b" uses same *vrfSegmentId* but different *vrfVlanId* as param_inst_50010. This is not allowed, as you cannot have the same segment ID being used for 2 different VLANs. This is the reason for the failure, and profile not being applied (rollback was successful).

The solution to this is having a unique SegmentId per VLAN (on the same switch).

Note: When running into the following error messages, always verify the needed licenses are properly installed (no grace-period licenses).

```
ERROR: config profile init in progress
ERROR: failed to find a free VSH session
```

3 Host Mobility via ARP and VDP

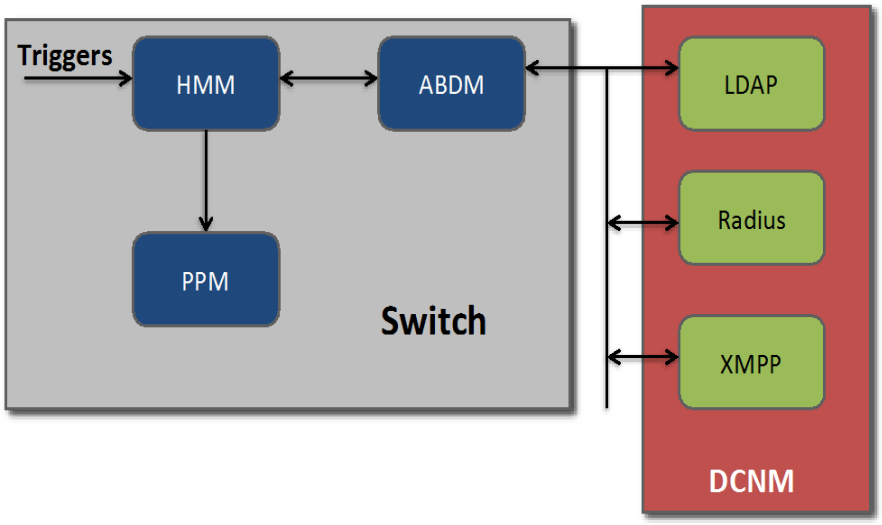
3.1 HMM and Auto-Configuration Profiles from ADBM

ADBM was created to allow access to multiple supported external databases for different types of DFA related information. The current types of information stored are server facing interface auto configuration, configuration profile contents, and cabling information.

The HMM process is responsible for host learning and initiating auto-configuration profiles download, and cleanup requests. A host's ARP requests is used by HMM to learn IP/MAC binding within a specific 802.1q tag and mobility-domain to associate the host to a vn-segment. This information is also used to query Cisco DCNM via LDAP to download auto-configuration profile on an as needed basis by each DFA leaf from ADBM. As such all leaf server-facing ports must accept 802.1q encapsulated frames to properly map the associated bridge domain and VN segment.

When a physical server or VM comes up, the connected leaf switch will detect the server or VM via either DHCP or ARP packets from the server. In turn, the DFA leaf will automatically configure the server-facing interface appropriately to allow packet forwarding to and from the detected server or VM.

Auto-configuration can optimize hardware and software resource utilization by selectively downloading network profiles only on leaves that require it. HMM learns and installs host routes into the local RIB as /32 routes. These routes are re-distributed into iBGP for reachability across the DFA fabric. Below is a block diagram of how these components interact.



HMM can trigger auto-configuration profile Instantiation via three methods:

- 802.1q packets (Any non-IP packet)
- VDP
- CLI

By default, leaf switches are configured for dynamic profiles instantiation. Both dot1Q tagged and VDP protocols use dynamic profiles based on queries to ADBM service.

```
dfa-n6k-leaf-1# show fabric database profile-map global

Flags: ? - Static profile not configured

Global Profile Map
(apply to all interfaces)

Map      Proto      VNI       DOT1Q     Flags  Profile Name
-------- ---------- --------- --------- ------ --------------------
global   ether-tag  default                    (dynamic)
global   ether-tag             default          (dynamic)
global   vdp        default                    (dynamic)
global   vdp                   default          (dynamic)
```

3.2 Dot1q Profile Instantiation

Host facing ports will utilize dot1q tagged frames in order to drive auto-configuration profile instantiation. In this example a host with MAC 0000.000c.00bb sends a dot1Q tagged frame, tagged with VLAN 100, to interface Ethernet 1/15 on a DFA leaf switch. The HMM process receives the frame via In-band Packet Manager, L2 network stack, and determines if no existing profile for VLAN 100 exists locally on the DFA leaf. HMM then queries ADBM service to search for a matching profile. Finally, HMM receives the auto-configuration profile and creates a VLAN to begin learning hosts within the new segment. The associated SVI (or BDI in case of Nexus 7000/7700) and VRF are also created

to support routing tenant traffic within or between VN segments. Any L2 traffic with proper dot1Q tagging can drive profile instantiation.

We can track the auto-profile instantiation steps in sequence. Be aware that NX-OS event-history outputs are in reverse chronological order. The host-learning event must be put into a "Profile Wait" state while the initial network and partition profiles query and auto-configuration process completes.

1. Interesting dot1q tagged Layer-2 frame is received by HMM process
2. Check global profile-map for ether-tag match will fail as using dynamic-profile
3. No local profile found in local database for given dot1q tag and mobility domain
4. Query ADBM database for profile matching mobility domain and dot1q tag
5. Leaf receives profile parameters from ADBM and installs in local profile database
6. Profile instantiation occurs on the local leaf

This sequence of events can be viewed in detail with the following event-history command.

```
dfa-n6k-leaf-1# show fabric forwarding internal event-history auto-config

Process auto-config logs of HMM

2015 Nov 11 19:13:25.823754 hmm [4660]: [4662]: Updating with keys: seg-id 0,bd 100 and Profile de-
faultNetworkUniversalEfProfile(instance_1) state 2 parameterized 1 recovery 0
2015 Nov 11 19:13:25.823748 hmm [4660]: [4662]: [100] Updating Parent state to Profile Wait
2015 Nov 11 19:13:25.823738 hmm [4660]: [4662]: Get instance - Could not find profile  with instance 0
2015 Nov 11 19:13:25.823730 hmm [4660]: [4662]: [100] Host entry has the include-vrf name sein-
feld:vandelay
2015 Nov 11 19:13:25.823726 hmm [4660]: [4662]: [100] ADBM Mark to-do query flags [01001010] : parti-
tion
2015 Nov 11 19:13:25.823711 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
VRF TLV: seinfeld:vandelay
2015 Nov 11 19:13:25.823703 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $include_vrfSegmentId=50000
2015 Nov 11 19:13:25.823695 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $mtuValue=
2015 Nov 11 19:13:25.823687 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $prefixLength=64
2015 Nov 11 19:13:25.823678 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $gatewayIpv6Address=2000:aaaa::1
2015 Nov 11 19:13:25.823671 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $vrfDhcp=
2015 Nov 11 19:13:25.823663 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $dhcpServerAddr=
2015 Nov 11 19:13:25.823655 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $netMaskLength=24
2015 Nov 11 19:13:25.823647 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $gatewayIpAddress=20.10.100.1
2015 Nov 11 19:13:25.823638 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $vrfName=seinfeld:vandelay
2015 Nov 11 19:13:25.823598 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $segmentId=30000
2015 Nov 11 19:13:25.823590 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $vlanId=100
2015 Nov 11 19:13:25.823580 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $netMaskLength=24
2015 Nov 11 19:13:25.823544 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Parameter: $gatewayIpAddress=20.10.100.1
```

```
2015 Nov 11 19:13:25.823498 hmm [4660]: [4662]: hmm_auto_config_syslog - msg_id 1, parent_entry
0x94c0c64 parent_entry type 1 vlan_id 0, cmd_data_vlan_id 0 key_bd_id 100 seg_id 0 cmd_dat
a_vni 0 key_seg_id 0
2015 Nov 11 19:13:25.823489 hmm [4660]: [4662]: Added active profile defaultNetworkUniversalEfProfile
with instance num 1 and instance name instance_def_100_1
2015 Nov 11 19:13:25.823474 hmm [4660]: [4662]: Adding Instance Entry 1
2015 Nov 11 19:13:25.823469 hmm [4660]: [4662]: Adding profile defaultNetworkUniversalEfProfile
2015 Nov 11 19:13:25.823455 hmm [4660]: [4662]: [100] ADBM Mark to-do query flags [01000010] : profile
content
2015 Nov 11 19:13:25.823447 hmm [4660]: [4662]: [100] ADBM Receive [ADD] Entry: PARENT, DB: NETWORK,
Profile Name: defaultNetworkUniversalEfProfile
2015 Nov 11 19:13:25.823423 hmm [4660]: [4662]: [100] ADBM Mark query done  [00000010] : network
2015 Nov 11 19:13:25.823385 hmm [4660]: [4662]: [100] ADBM Recv results (1) [00000010]:[00000000]------
------------
2015 Nov 11 19:13:25.823310 hmm [4660]: [4662]: Decrement outstanding ADBM request (1/50) -> (0/50)
2015 Nov 11 19:13:25.811748 hmm [4660]: [4669]: Increment outstanding ADBM request (0/50) -> (1/50)

2015 Nov 11 19:13:25.811713 hmm [4660]: [4669]: [100] Updating Host state change [Err: No profile] =>
[Profile Wait] (HOST-VLAN Ethernet1/15)
2015 Nov 11 19:13:25.811707 hmm [4660]: [4669]: [100] Updating Host state to Profile Wait
2015 Nov 11 19:13:25.811696 hmm [4660]: [4669]: Get instance - Could not find profile  with instance 0
2015 Nov 11 19:13:25.811692 hmm [4660]: [4669]: [100] Updating Parent state to Profile Wait
2015 Nov 11 19:13:25.811688 hmm [4660]: [4669]: [100] ADBM Sent request for database type 1 and got
req_id 1
2015 Nov 11 19:13:25.811665 hmm [4660]: [4669]: called adbm_send_msg and got result 0
2015 Nov 11 19:13:25.811517 hmm [4660]: [4669]: Add ADBM attribute: vlan_id: 100
2015 Nov 11 19:13:25.811513 hmm [4660]: [4669]: Add ADBM attribute: mobility domain: md0
2015 Nov 11 19:13:25.811449 hmm [4660]: [4669]: ret_val is false so use global mobility domain name,
md0
2015 Nov 11 19:13:25.811343 hmm [4660]: [4669]: [100] ADBM Run  network/bl-dci query
2015 Nov 11 19:13:25.811338 hmm [4660]: [4669]: [100] ADBM Run  queries (0)
[00000010]:[00000000]==================
2015 Nov 11 19:13:25.811285 hmm [4660]: [4662]: Get instance - Could not find profile  with instance 0
2015 Nov 11 19:13:25.811279 hmm [4660]: [4662]: 100:0000.000c.00bb Sent to ADBM to get profile name
2015 Nov 11 19:13:25.811236 hmm [4660]: [4662]: Signaling the profile writer thread to launch more to a
2015 Nov 11 19:13:25.811228 hmm [4660]: [4662]: [100] ADBM Init query flags [00000010]
2015 Nov 11 19:13:25.811219 hmm [4660]: [4662]: [100] ADBM Mark to-do query flags [00000010] : network
2015 Nov 11 19:13:25.811212 hmm [4660]: [4662]: [100] ADBM Init query flags [00000000] (clear)
2015 Nov 11 19:13:25.811205 hmm [4660]: [4662]: [100] (Re)started MAC Aging timer: 30 minutes

2015 Nov 11 19:13:25.811178 hmm [4660]: [4662]: [100] Updating Host state change [Err: No profile] =>
[Err: No profile] (HOST-VLAN Ethernet1/15)
2015 Nov 11 19:13:25.811169 hmm [4660]: [4662]: [100] Added host with initial state [Err: No profile
(0)]
2015 Nov 11 19:13:25.811156 hmm [4660]: [4662]: Client is expecting a reply
2015 Nov 11 19:13:25.811147 hmm [4660]: [4662]: [100] Adding single host with vni:0 ifindex 0x1a00e000
encap:100
2015 Nov 11 19:13:25.811130 hmm [4660]: [4662]: Adding new entry for ifIndex Ethernet1/15 vlan 100
2015 Nov 11 19:13:25.811111 hmm [4660]: [4662]: Updated auto_config vlan 100 seg_id 0 in PSS
2015 Nov 11 19:13:25.810970 hmm [4660]: [4662]: Adding with keys: seg-id 0,bd 100
2015 Nov 11 19:13:25.810952 hmm [4660]: [4662]: Adding active host on interface Ethernet1/15, vlan 100
(encap 100), vni 0, mac 0000.000c.00bb
2015 Nov 11 19:13:25.810910 hmm [4660]: [4662]: 100:0000.000c.00bb Got Dynamic Profile Name (null)
2015 Nov 11 19:13:25.810901 hmm [4660]: [4662]: Match the global map for ether-tag trigger 100:0
2015 Nov 11 19:13:25.810886 hmm [4660]: [4662]: Did not match the inherited map for ether-tag trigger
100:0
2015 Nov 11 19:13:25.809553 hmm [4660]: [4662]: 100:0000.000c.00bb Got ARP request on interface Ether-
net1/15
```

Using the **config-profile history** command, DFA administrators can verify which profiles were Applied and Un-applied from a DFA leaf switch. Below we can see the partition and network profiles were applied locally to *dfa-n6k-leaf-1*.

```
dfa-n6k-leaf-1# show system internal config-profile history

===============================================================================
Inst/Param List Name   : param_inst_50000/param_list
Include List/Inst Name : /
Refresh Profile Name   :
Refresh List/Inst Name : /
Operation Type         : Apply
Time Requested         : 11/11/15 19:13:27.230765
Time Responded         : 11/11/15 19:13:27.713719
Rollback Status        : Rollback not done
Configuration Type     : Auto Config
===============================================================================
Inst/Param List Name   : instance_def_100_1/param_list_name_instance_def_100_1
Include List/Inst Name : include_param_list_name_seinfeld:vandelay_3/seinfeld:vandelay_3
Refresh Profile Name   :
Refresh List/Inst Name : /
Operation Type         : Apply
Time Requested         : 11/11/15 19:13:25.829609
Time Responded         : 11/11/15 19:13:27.935154
Rollback Status        : Rollback not done
Configuration Type     : Auto Config
===============================================================================
```

The local leaf fabric database can be queried to verify profile details, such as which ports a profile is actively applied on.

```
dfa-n6k-leaf-1# show fabric database host detail dot1q 100

instance_index 1
Got Local originated vlan type trigger at 19:13:25
Number of associated interfaces: 1
The profile will be checked for aging in 1710 seconds
Sent to Database Manager at 19:13:25
Received Parameters from Database Manager at 19:13:25
Displaying parameters for profile defaultNetworkUniversalEfProfile and instance instance_def_100_1
parameter 0: $gatewayIpAddress=20.10.100.1
parameter 1: $netMaskLength=24
parameter 2: $vlanId=100
parameter 3: $segmentId=30000
parameter 4: $vrfName=seinfeld:vandelay
parameter 5: $gatewayIpAddress=20.10.100.1
parameter 6: $netMaskLength=24
parameter 7: $dhcpServerAddr=
parameter 8: $vrfDhcp=
parameter 9: $gatewayIpv6Address=2000:aaaa::1
parameter 10: $prefixLength=64
parameter 11: $mtuValue=
parameter 12: $include_vrfSegmentId=50000
parameter 13: $vlanId=100
parameter 14: $asn=65000
Sent Apply to Configuration Manager at 19:13:25
Completed executing all commands at 19:13:27
Sent to vPC peer at 19:13:27
Completed executing all commands on vPC peer at 19:13:30
```

```
Displaying Data Snooping Ports
Interface      Encap      Flags State
Eth1/15        100        L     Profile Active
```

Subsequent to the auto-configuration profile being pulled by the DFA leaf, it can setup the appropriate VLAN and SVI configuration to support traffic in the new vn-network segment. Below you can see the L2 and L3 information for vlan100, and what VRF it is associated with.

```
dfa-n6k-leaf-1# show vlan id 100

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
100  VLAN0100                         active    Po1, Po10, Po20, Po30, Po40
                                                Po50, Po100, Po110, Eth1/1
                                                Eth1/2, Eth1/3, Eth1/4, Eth1/5
                                                Eth1/6, Eth1/7, Eth1/8, Eth1/9
                                                Eth1/10, Eth1/13, Eth1/14
                                                Eth1/15, Eth1/16, Eth1/17
                                                Eth1/18, Eth1/19, Eth1/33
                                                Eth1/34, Eth1/35, Eth1/36
                                                Eth2/1/1, Eth2/2/1, Eth2/2/2
                                                Eth2/2/3, Eth2/2/4, Eth2/3
                                                Eth2/4

VLAN Type  Vlan-mode
---- ----- ----------
100  enet  FABRICPATH

Primary  Secondary  Type              Ports
-------  ---------  ---------------   -----------------------------------------


dfa-n6k-leaf-1# show spanning-tree vlan 100

VLAN0100
  Spanning tree enabled protocol rstp
  Root ID    Priority    32868
             Address     c84c.75fa.6000
             This bridge is the root
             Hello Time  2  sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32868  (priority 32768 sys-id-ext 100)
             Address     c84c.75fa.6000
             Hello Time  2  sec  Max Age 20 sec  Forward Delay 15 sec

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Eth1/15          Desg FWD 4         128.143  P2p


dfa-n6k-leaf-1# show running-config interface Vlan100 expand-port-profile

!Command: show running-config interface Vlan100 expand-port-profile
!Time: Wed Nov 11 19:15:45 2015
```

```
version 7.2(1)N1(1)

interface Vlan100
  no shutdown
  vrf member seinfeld:vandelay
  no ip redirects
  ip address 20.10.100.1/24 tag 12345
  ipv6 address 2000:aaaa::1/64 tag 12345
  no ipv6 redirects
  fabric forwarding mode proxy-gateway


dfa-n6k-leaf-1# show interface vlan 100

Vlan100 is up, line protocol is up
  Hardware is EtherSVI, address is  002a.6aa1.0c81
  Internet Address is 20.10.100.1/24
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec
```

A DFA fabric administrator can also determine how many profiles have been requested and dispatched to a given DFA leaf using the fabric statistics command.

```
dfa-n6k-leaf-1# show fabric database statistics
DB-Type         Requests    Dispatched  Not dispatched  Re-dispatched
-------------------------------------------------------------------
network                1             1               0              0
cabling                0             0               0              0
profile                0             0               0              0
partition              1             1               0              0
bl-dci                 0             0               0              0
-------------------------------------------------------------------
TOTAL                  2             2               0              0

Per Database stats:
 T Prot Server/DB                       Reqs     OK  NoRes    Err  TmOut   Pend
-------------------------------------------------------------------------------
*ne ldap dcg-dcnm.cisco.com                1      1      0      0      0      0
*pr ldap dcg-dcnm.cisco.com                0      0      0      0      0      0
*pa ldap dcg-dcnm.cisco.com                1      1      0      0      0      0
Legend:
  T-Type (ne-network, ca-cabling, pr-profile, pa-partition, bl-bl-dci)
  *-Active Server
```

3.3 HMM Triggered via ARP

3.3.1 Dot1q Host Learning Event

Subsequent to partition and network profile instantiation, a host generated ARP request can trigger HMM to install a HMM host entry. This entry can then be re-distributed into iBGP, in a given VRF/Partition, for accessibility across the fabric without the need to flood ARP across the DFA fabric. The below event-history command shows the HMM database populated with a MAC/IP binding for a specific partition. “AM” in the output below represents the Adjacency Manager process.

```
dfa-n6k-leaf-1# show fabric forwarding internal event-history events
```

```
Process Event logs of HMM
2015 Nov 11 19:17:04.680217 hmm [4660]: [4662]: (seinfeld:vandelay) MAC: 0000.000b.00bb old_port: -
(0x00000000), new_port: Ethernet1/15(0x1a00e000), vlan_id: 0x00000064, bd_id: 100, is_d
el: 0, Svi: Vlan100(3)
2015 Nov 11 19:17:04.680138 hmm [4660]: [4662]: Got a MAC notification from L2FM for 1 adjacencies
2015 Nov 11 19:17:04.580764 hmm [4660]: [4673]: (seinfeld:vandelay) [IPv4] tbl_ctx created, table base,
initial state Up
2015 Nov 11 19:17:04.580694 hmm [4660]: [4673]: (seinfeld:vandelay) [IPv4] Enabling tbl_ctx
2015 Nov 11 19:17:04.580659 hmm [4660]: [4673]: (seinfeld:vandelay) Vrf (Id: 5) created, state Up
2015 Nov 11 19:17:04.580640 hmm [4660]: [4673]: (seinfeld:vandelay) Updated Vrf (Id: 5), state Up
2015 Nov 11 19:17:04.580603 hmm [4660]: [4673]: (seinfeld:vandelay) [IPv4] Received AM notification for
Host 20.10.100.71/32, mac 0000.000b.00bb, svi Vlan100, l2_port Ethernet1/15, flags
 0x00000000
2015 Nov 11 19:17:04.580546 hmm [4660]: [4673]: Received AM notification with 1 entries), new_port:
Ethernet1/15(0x1a00e000), vlan_id: 0x00000064, bd_id: 100
```

The local HMM host database is populated with host MAC/IP binding under the partition VRF. A DFA administrator can review the summary of host counts and detailed output listing learned hosts' information. The new host entry, comprised of IP and MAC pair, will be learned against a specific edge interface for an Anycast SVI previously applied by auto-config profile instantiation.

```
dfa-n6k-leaf-1# show fabric forwarding host-db vrf seinfeld:vandelay

HMM Host DB information for VRF seinfeld:vandelay
VRF Id                          : 5
VRF state                       : Up
VRF reason                      : --
VNI id                          : 50000
Refcount                        : 1
Conversational Learning         : No

    Information for address family IPv4 in VRF seinfeld:vandelay
    Table Id                    : 0x00000003
    Table State                 : Up
    Refcount                    : 1
    Local hosts          Remote hosts         Aggregates
    1                    0                    0

dfa-n6k-leaf-1# show fabric forwarding ip local-host-db vrf seinfeld:vandelay

HMM host IPv4 routing table information for VRF seinfeld:vandelay
Status: *-valid, x-deleted, c-cleaned in 00:03:00

    Host                 MAC Address        SVI        Flags      Physical Interface
*   20.10.100.71/32      0000.000b.00bb     Vlan100    0x20201    Ethernet1/15
```

Locally, the leaf switch RIB is updated with the /32 route learned by HMM. In this way HMM is a client protocol of the Routing Information Base in NX-OS, just as ARP, OSPF, EIGRP, or any other routing protocol would be.

```
dfa-n6k-leaf-1# show ip route vrf seinfeld:vandelay

IP Route Table for VRF "seinfeld:vandelay"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
```

```
'%<string>' in via output denotes VRF <string>

20.10.100.0/24, ubest/mbest: 1/0, attached
    *via 20.10.100.1, Vlan100, [0/0], 00:07:58, direct, tag 12345,
20.10.100.1/32, ubest/mbest: 1/0, attached
    *via 20.10.100.1, Vlan100, [0/0], 00:07:58, local, tag 12345,
20.10.100.11/32, ubest/mbest: 1/0
    *via 10.1.10.123%default, [200/0], 00:07:47, bgp-65000, internal, tag 65000, segid 50000
20.10.100.70/32, ubest/mbest: 1/0
    *via 10.1.10.123%default, [200/0], 00:07:47, bgp-65000, internal, tag 65000, segid 50000
20.10.100.71/32, ubest/mbest: 1/0, attached
    *via 20.10.100.71, Vlan100, [190/0], 00:04:21, hmm

dfa-n6k-leaf-1# show run bgp | section "vrf seinfeld:vandelay"

  vrf seinfeld:vandelay
    address-family ipv4 unicast
      redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
      redistribute direct route-map FABRIC-RMAP-REDIST-SUBNET
      maximum-paths ibgp 2
    address-family ipv6 unicast
      redistribute hmm route-map FABRIC-RMAP-REDIST-V6HOST
      redistribute direct route-map FABRIC-RMAP-REDIST-SUBNET
      maximum-paths ibgp 2

dfa-n6k-leaf-1# show ip bgp vrf seinfeld:vandelay 20.10.100.70

BGP routing table information for VRF seinfeld:vandelay, address family IPv4 Unicast
BGP routing table entry for 20.10.100.70/32, version 8
Paths: (1 available, best #1)
Flags: (0x08061a) on xmit-list, is in urib, is best urib route
  vpn: version 53, (0x100002) on xmit-list
Multipath: iBGP
  Advertised path-id 1, VPN AF advertised path-id 1
  Path type: internal, path is valid, is best path
             Imported from 10.1.10.123:5:20.10.100.70/32
  AS-Path: NONE, path sourced internal to AS
    10.1.10.123 (metric 0) from 10.1.10.105 (10.1.10.105)
      Origin incomplete, MED 0, localpref 100, weight 0
      Received path-id 1
      Extcommunity:
          RT:65000:50000
          SOO:0.0.0.1:1
          COST:pre-bestpath:96:1610612736 transitive
          VNID:50000
      Originator: 10.1.10.123 Cluster list: 10.1.10.105

  VRF advertise information:
  Path-id 1 not advertised to any peer

  VPN AF advertise information:
  Path-id 1 not advertised to any peer
```

Looking at the below table of a remote DFA leaf switch, we can see the HMM host entry is advertised by iBGP as a /32 route across the fabric. This process allows for reachability of individual /32 host entries.

```
dfa-n77k-1-dfa-n77-leaf-1# show ip bgp vrf seinfeld:vandelay
```

```
BGP routing table information for VRF seinfeld:vandelay, address family IPv4 Uni
cast
BGP table version is 453322, local router ID is 20.10.100.1
Status: s-suppressed, x-deleted, S-stale, d-dampened, h-history, *-valid, >-best
Path type: i-internal, e-external, c-confed, l-local, a-aggregate, r-redist, I-i
njected
Origin codes: i - IGP, e - EGP, ? - incomplete, | - multipath, & - backup

   Network            Next Hop            Metric     LocPrf     Weight Path
* i20.10.100.0/24     10.1.10.122              0        100          0 ?
* i                   10.1.10.121              0        100          0 ?
* i                   10.1.10.124              0        100          0 ?
*>r                   0.0.0.0                  0        100      32768 ?
*>r20.10.100.11/32    0.0.0.0                  0        100      32768 ?
*>r20.10.100.70/32    0.0.0.0                  0        100      32768 ?
*>i20.10.100.71/32    10.1.10.121              0        100          0 ?
```

3.3.2 HMM Handing a Host Move Event

Host movement across a DFA fabric can be detected by Layer-2, non-IP traffic or IP ARP traffic independently. Some hosts may generate Layer-2 non-IP traffic and IP traffic while other hosts only generate one or the other at a given period of time.

Host movement events are tracked by two separate but related processes, *MAC move notification* and *Adjacency move notification*. If Fabricpath learns the MAC remotely, but there is no ARP/Adjacency update, HMM will then update the old DFA leaf switch with the new MAC location via a notification from the Layer 2 Feature Manager (L2FM) process. If IP traffic is routed across the fabric using an existing /32 route, then it is forwarded to the old DFA leaf. As a result, the DFA leaf will switch traffic via Layer-2 FabricPath back across the fabric to the new host location that advertised the move via L2FM, but the updated MAC table learns the FabricPath Switch ID. This sub-optimal hair pinning of traffic across the DFA fabric will be resolved or avoided by prompt receipt of an ARP frame from the new DFA leaf.

When the new DFA leaf switch snoops an ARP packet from the moved host, a new HMM host entry is created. This new route will be re-distributed across the DFA fabric and trigger a flush of the host entry on the old leaf, where HMM previously learned the host location. Subsequently, all routed traffic to this host IP will be forwarded directly to the new DFA leaf next-hop.

When a locally learned host HMM IP/MAC binding is seen across the fabric on a remote DFA leaf the stale entry must be flushed from HMM and withdrawn from iBGP to support host mobility. It is important that hosts re-ARP after a move event for quick propagation of the correct route entry across the DFA fabric.

Initially we can see that host 20.10.100.11 is locally learned by HMM on port-channel 10.

```
dfa-n6k-leaf-1# show fabric forwarding ip local-host-db vrf seinfeld:vandelay 20.10.100.11/32

HMM routing table information for VRF seinfeld:vandelay, address family IPv4
HMM routing table entry for 20.10.100.11/32
Hosts: (1 available)

  Host type: Local(Flags: 0x20201), in Rib
  mac: 0000.0000.000a, svi: Vlan100, bd: 100, phy_intf: port-channel10
```

```
dfa-n6k-leaf-1# show mac address-table vlan 100

Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since last seen,+ - primary entry using vPC Peer-Link
   VLAN     MAC Address      Type      age     Secure NTFY   Ports/SWID.SSID.LID
---------+-----------------+--------+---------+------+----+------------------
* 100      0000.0000.000a    dynamic   20         F    F  Po10
* 100      002a.6a78.9c3c    static    0          F    F  122.0.3
* 100      2020.0000.00aa    static    0          F    F  sup-eth2
```

As soon as the leaf switch detects the MAC has moved, L2 Forwarding Manager (L2FM) will send a notification to HMM in order to update the local HMM entry for IPv4 and IPv6 as necessary. Below you can verify that the local HMM entry still temporarily exists, but points to the remote FP switch-id of the leaf the host moved to

```
dfa-n6k-leaf-1# show fabric forwarding ip local-host-db vrf seinfeld:vandelay 20.10.100.11/32
HMM routing table information for VRF seinfeld:vandelay, address family IPv4
HMM routing table entry for 20.10.100.11/32
Hosts: (1 available)

  Host type: Local(Flags: 0x20201), in Rib
  mac: 0000.0000.000a, svi: Vlan100, bd: 100, phy_intf: Gateway Port-Channel1:250
```

```
dfa-n6k-leaf-1# show mac address-table vlan 100
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since last seen,+ - primary entry using vPC Peer-Link
   VLAN     MAC Address      Type      age     Secure NTFY   Ports/SWID.SSID.LID
---------+-----------------+--------+---------+------+----+------------------
* 100      0000.0000.000a    dynamic   0          F    F  250.12.65535
* 100      0000.0000.000b    dynamic   70         F    F  Po20
* 100      002a.6a78.9c3c    static    0          F    F  122.0.3
* 100      2020.0000.00aa    static    0          F    F  sup-eth2
```

Finally, a notification from ARP Adjacency Manager triggers HMM to flush the local entry for 20.10.100.11. This completes the move operation. Typically, these two operations happen within the same second for a production DFA environment. Below you can see the arp table with the respective host, as well as the events that verify the table being populated.

```
dfa-n6k-leaf-1# show ip arp vrf seinfeld:vandelay

Flags: * - Adjacencies learned on non-active FHRP router
       + - Adjacencies synced via CFSoE
       # - Adjacencies Throttled for Glean
       D - Static Adjacencies attached to down interface

IP ARP Table for context seinfeld:vandelay
Total number of entries: 3
Address         Age       MAC Address     Interface
20.10.100.11    00:01:42  0000.0000.000a  Vlan100
20.10.100.12    00:01:18  0000.0000.000b  Vlan100
```

```
dfa-n6k-leaf-1# show fabric forwarding internal event-history events

Process Event logs of HMM
2015 Nov 11 20:38:19.240136 hmm [4660]: [4662]: (seinfeld:vandelay) MAC: 0000.0000.000a old_port: port-
channel10(0x16000009), new_port: Gateway Port-Channel1:250(0x240010fa), vlan_id: 0x00000064, bd_id:
100, is_del: 0, Svi: V
lan100(3)
2015 Nov 11 20:38:19.240068 hmm [4660]: [4662]: Got a MAC notification from L2FM for 1 adjacencies

dfa-n6k-leaf-1# show fabric forwarding internal event-history events

Process Event logs of HMM
2015 Nov 11 20:41:18.230522 hmm [4660]: [4673]: (seinfeld:vandelay) [IPv4] Received AM notification for
Host 20.10.100.11/32, mac 0000.0000.0000, svi Vlan100, l2_port Gateway Port-Channel1:250, flags
0x00000000
2015 Nov 11 20:41:18.230465 hmm [4660]: [4673]: Received AM notification with 1 entries

fa-n6k-leaf-1# show fabric forwarding ip local-host-db vrf seinfeld:vandelay 20.10.100.11/32

       [no output]


dfa-n6k-leaf-1# show ip route vrf seinfeld:vandelay

IP Route Table for VRF "default"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

10.1.10.0/24, ubest/mbest: 1/0, attached
    *via 10.1.10.121, Vlan10, [0/0], 03:32:59, direct
10.1.10.121/32, ubest/mbest: 1/0, attached
    *via 10.1.10.121, Vlan10, [0/0], 03:32:59, local
dfa-n6k-leaf-1# show ip route  vrf seinfeld:vandelay
IP Route Table for VRF "seinfeld:vandelay"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

20.10.100.0/24, ubest/mbest: 1/0, attached
    *via 20.10.100.1, Vlan100, [0/0], 00:57:48, direct, tag 12345,
20.10.100.1/32, ubest/mbest: 1/0, attached
    *via 20.10.100.1, Vlan100, [0/0], 00:57:48, local, tag 12345,
20.10.100.11/32, ubest/mbest: 1/0
    *via 10.1.10.123%default, [200/0], 00:00:27, bgp-65000, internal, tag 65000, segid 50000
20.10.100.12/32, ubest/mbest: 1/0, attached
    *via 20.10.100.12, Vlan100, [190/0], 00:56:55, hmm
20.10.100.70/32, ubest/mbest: 1/0
    *via 10.1.10.123%default, [200/0], 00:21:31, bgp-65000, internal, tag 65000, segid 50000
```

3.3.3 Dot1q Profile Aging and Cleanup

HMM uses a Profile Aging timer, 30 minutes by default, to check if the VLAN/BD has any MAC entries learned or not. This timer based approach prevents rapid profile auto-config churn for networks and partitions with intermittent traffic sources. The aging and cleanup process allows DFA leaf switches to only carry profile information that is needed for a

given leaf switch, thereby optimizing resources used by each leaf switch. Also note that removing a network or partition profile from Cisco DCNM does not cause the profile to be immediately removed from a DFA leaf switch.

```
dfa-n6k-leaf-1# show fabric database host summary
Number of instances applied :    1
Number of VDP hosts         :    0
Recovery Timeout Value      :    30 minutes
Cleanup Timeout Value       :    15 minutes
VDP Add Suppression Timeout :    2 minutes
Profiles checked for aging  :    30 minutes
[Option] auto-id support    :    system_auto_loopbackId,system_auto_backboneIpAddress

dfa-n6k-leaf-1# show fabric database host detail dot1q 100
instance_index 3
Got Local originated vlan type trigger at 19:43:57
Number of associated interfaces: 1
The profile will be checked for aging in 1690 seconds
Sent to Database Manager at 19:43:57
Received Parameters from Database Manager at 19:43:57
Displaying parameters for profile defaultNetworkUniversalEfProfile and instance instance_def_100_3
parameter 0: $gatewayIpAddress=20.10.100.1
parameter 1: $netMaskLength=24
parameter 2: $vlanId=100
parameter 3: $segmentId=30000
parameter 4: $vrfName=seinfeld:vandelay
parameter 5: $gatewayIpAddress=20.10.100.1
parameter 6: $netMaskLength=24
parameter 7: $dhcpServerAddr=
parameter 8: $vrfDhcp=
parameter 9: $gatewayIpv6Address=2000:aaaa::1
parameter 10: $prefixLength=64
parameter 11: $mtuValue=
parameter 12: $include_vrfSegmentId=50000
parameter 13: $vlanId=100
parameter 14: $asn=65000
Sent Apply to Configuration Manager at 19:43:57
Completed executing all commands at 19:43:58
Sent to vPC peer at 19:43:58
Completed executing all commands on vPC peer at 19:43:58
Displaying Data Snooping Ports
Interface      Encap      Flags State
Po10           100        L     Profile Active
```

When a MAC address is flushed or aged from the MAC address table, L2FM will notify HMM of the delete notification. The following command can help verify these events.

```
dfa-n6k-leaf-1# show fabric forwarding internal event-history trace | grep "Received MAC delete notifi-
cation"

2015 Nov 11 20:36:17.753363 hmm [4660]: [4662]: Received MAC delete notification for MAC:
0000.0000.000b old_port: port-channel20(0x16000013), new_port: port-channel20(0x16000013), vlan_id:
0x00000064, bd_id: 100
2015 Nov 11 20:23:32.801591 hmm [4660]: [4662]: Received MAC delete notification for MAC:
0000.0000.000a old_port: port-channel10(0x16000009), new_port: port-channel10(0x16000009), vlan_id:
0x00000064, bd_id: 100
```

During the next Profile Aging timer interval, HMM identifies the VLAN is empty and eligible for deletion. The profile cleanup timer, 15 minutes by default, is started to track profile cleanup and removal.

```
dfa-n6k-leaf-1# show fabric forwarding internal event-history events | grep "deleted"

2015 Nov 11 21:06:12.953086 hmm [4660]: [4667]: (seinfeld:vandelay) vrf_ctx cleanup completed, deleted
2015 Nov 11 21:06:12.953080 hmm [4660]: [4667]: (seinfeld:vandelay) [IPv4] Table cleanup complete, de-
leted tid 3


dfa-n6k-leaf-1#  show fabric database host detail dot1q 100

instance_index 3
Got Unknown originated vlan type trigger at 19:43:57
Number of associated interfaces: 0
Profile will be un-applied in 450 seconds
Sent to Database Manager at 19:43:57
Received Parameters from Database Manager at 19:43:57
Displaying parameters for profile defaultNetworkUniversalEfProfile and instance instance_def_100_3
parameter 0: $gatewayIpAddress=20.10.100.1
parameter 1: $netMaskLength=24
parameter 2: $vlanId=100
parameter 3: $segmentId=30000
parameter 4: $vrfName=seinfeld:vandelay
parameter 5: $gatewayIpAddress=20.10.100.1
parameter 6: $netMaskLength=24
parameter 7: $dhcpServerAddr=
parameter 8: $vrfDhcp=
parameter 9: $gatewayIpv6Address=2000:aaaa::1
parameter 10: $prefixLength=64
parameter 11: $mtuValue=
parameter 12: $include_vrfSegmentId=50000
parameter 13: $vlanId=100
parameter 14: $asn=65000
Sent Apply to Configuration Manager at 19:43:57
Completed executing all commands at 19:43:58
Sent to vPC peer at 21:08:55
Completed executing all commands on vPC peer at 21:08:55
Got no hosts for this profile
```

After the profile cleanup timer expires the profile is flushed and removed from the local DFA leaf. Use the following command(s) to verify the same.

```
dfa-n6k-leaf-1# show fabric forwarding internal event-history events | grep "deleted"

2015 Nov 11 21:23:58.628864 hmm [4660]: [4662]: Deleted include-vrf with name seinfeld:vandelay from
PSS
2015 Nov 11 21:23:58.110944 hmm [4660]: [4662]: Deleted sviinfo SDB for svi: Vlan100

2015 Nov 11 21:06:12.953086 hmm [4660]: [4667]: (seinfeld:vandelay) vrf_ctx cleanup completed, deleted
2015 Nov 11 21:06:12.953080 hmm [4660]: [4667]: (seinfeld:vandelay) [IPv4] Table cleanup complete, de-
leted tid 3


dfa-n6k-leaf-1# show fabric database host detail dot1q 100

Unable to find an entry
```

```
dfa-n6k-leaf-1# show system internal config-profile history

===============================================================================
Inst/Param List Name   : param_inst_50000/param_list
Include List/Inst Name : /
Refresh Profile Name   :
Refresh List/Inst Name : /
Operation Type         : Un-Apply
Time Requested         : 11/11/15 21:23:58.067647
Time Responded         : 11/11/15 21:23:58.466221
Rollback Status        : Rollback not done
Configuration Type     : Auto Config
===============================================================================
Inst/Param List Name   : instance_def_100_3/param_list_name_instance_def_100_3
Include List/Inst Name : include_param_list_name_seinfeld:vandelay_3/seinfeld:vandelay_3
Refresh Profile Name   :
Refresh List/Inst Name : /
Operation Type         : Un-Apply
Time Requested         : 11/11/15 21:23:56.724579
Time Responded         : 11/11/15 21:23:58.626641
Rollback Status        : Rollback not done
Configuration Type     : Auto Config
===============================================================================
```

3.4 HMM Triggered via VDP

The Virtual Station Interface (VSI) Discovery and Configuration Protocol (VDP) was defined in the IEEE 802.1Qbg and is now integrated in 802.1Q-2014 IEEE standard). _VDP serves as a reliable first-hop protocol between Nexus 1000V switches and their adjacent leaf switch nodes in the DFA architecture. VDP is used to communicate the presence of end-host Virtual Machines (VMs) Interfaces and their network parameters, such as the network a VM interface belongs to. This allows on-demand network resources instantiation. A network in this context can be an 802.1Q VLAN (VLAN-based VDP) or a Layer-2 segment (segment-based VDP), in which case the leaf switch will dynamically allocate a VLAN and instruct the Nexus 1000V switch to place the VM traffic in this VLAN. The following information will focus around segment-based VDP.

Key Terms:

- **EVB**: Edge Virtual Bridging, described in IEEE 802.1Q-2014 standard, enables coordinated configuration and management of bridge services for virtual stations in a network.
- **Virtual Station Interface (VSI)**: An internal point-to-point Ethernet LAN that connects a virtual machine to a port of a virtual switch, such as the Nexus 1000V switch. A VSI is logically referred to as a VSI ID.
- **Virtual Station Interface (VSI) Discovery and Configuration (VDP)**: A protocol that supports the association of a VSI with a bridge port.
- **ECP** : Edge Control Protocol, transport protocol for VDP PDUs providing a reliable transmission
- **VDP Station**: A system that initiates VDP exchange to signal its presence and connection needs.
- **VDP Bridge**: The edge bridge to which the VDP Stations are attached.

VSI Discovery and Configuration Protocol:

VDP enables the association (registration) and de-association of virtual machine interfaces (VSIs) with a server-facing leaf switch interface.
VDP makes uses of TLVs (type/length/value triplets) to carry information between a VDP Station and a VDP bridge. The standard defines 3 sets of TLVs :

- The VSI manager TLV allows to address a given VSI manager or VSI database. This TLV is present in every VDP PDUs (VDPUs) and must be the first TLV. A value of 0 means the station does not know what VSI manager to use.
- VDP association TLVs whose types are Pre-Associate, Pre-Associate with resource reservation, Associate and De-associate. The figure below illustrates the format of the association TLVs. Of interest is the filter info field which is a subset of VLAN ID, MAC address, group ID (segment ID) and IP address, the combination of which is defined by the filter info format field.

TLV type (7 bits)	TLV length (9 bits)	Status (1 byte)	VSI Type ID (3 bytes)	VSI Type Version (1 byte)	VSIID format (1 byte)	VSIID (16 bytes)	Filter Info format (1 byte)	Filter Info (M bytes)

- Organizationally defined TLVs, allows extending the protocol to carry organization specific information. Nexus 1000V and DFA leaf switches make use of these files to carry Virtual Machines names.

VSI Discovery and Configuration Protocol Sequence:

1. When a VM is activated, the VDP station (Nexus 1000V switch) passes the network information to the DFA leaf switch through a VDP association request. The network information is carried in the filter info field of the association TLV. In case of segmentation-based VDP, the group ID is populated with the value of the network segment, and the VLAN ID is set to a null value indicating the VDP station does not yet know which locally significant VLAN ID is associated with the Layer-2 segment.
2. After receiving the association request, the leaf switch extracts the network information and automatically configures and attaches a VLAN value to the segment ID.
3. The leaf switch then sends a response to the Cisco Nexus 1000V Series switch after the *filter info* field is modified with the new VLAN information. The Cisco Nexus 1000V applies this VLAN value in the dot1q encapsulation of packets for that VM.
4. After a VM is successfully associated, the Nexus 1000V switch periodically sends the association TLV to the leaf switch for a state refresh.

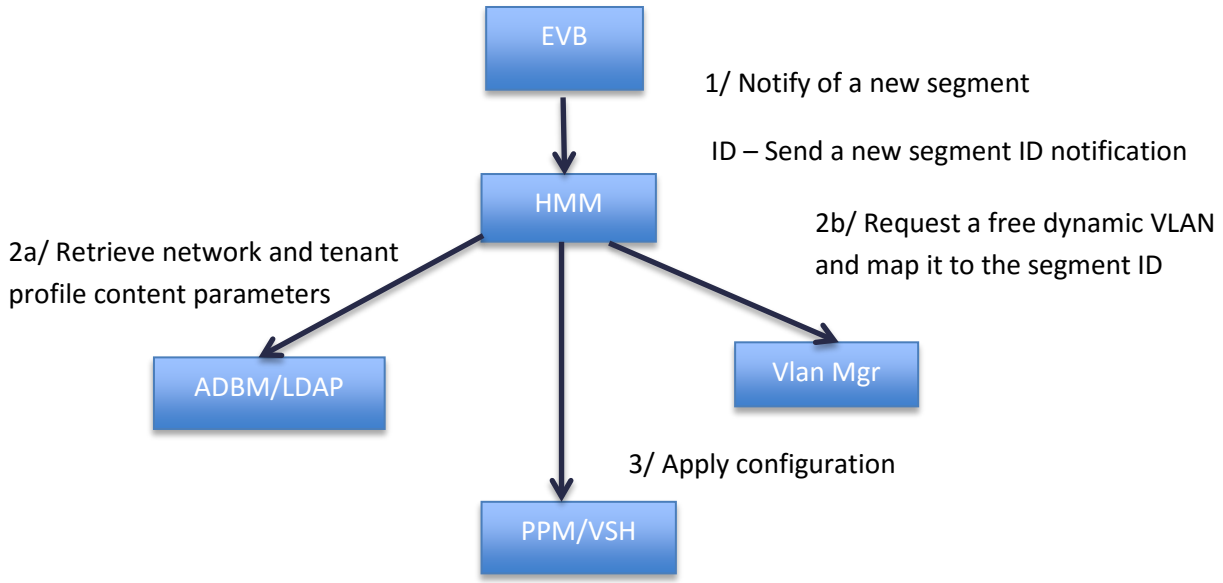
Timers:

An EVB protocol timer in 802.1Q-2014 standard is expressed as a timer exponent whose definition is a value between 0 and 31, representing a positive integer for the exponent of 2, which forms the multiplier of 10 microseconds to give the timer value. For example, a value of 4 means the associate timer value equals 2^4 x 10 µs, or 160 µs.

The table below indicates some default exponent values used on DFA leaf switches and the Nexus 1000V switch:

Exponent value	**Approximate timer value**
14 (default ECP retransmission value)	163 milliseconds
20 (default keepalive value on Nexus1000V switch)	10 seconds
25 (default resource-wait value on leaf switch)	335 seconds
27 (default keepalive value on leaf switch)	22 minutes

3.5 VDP Profile Instantiation (High-level overview)



The major steps involved in the segment-based profile instantiation are:

- An association request is made for a given segment.
- If the segment's configuration has not yet been instantiated, the HMM process retrieves the necessary information from the LDAP database via ADBM process.
- In parallel, the HMM requests a free dynamic VLAN from the VLAN manager process
- When profile contents and parameters are available, the HMM process requests the port-profile process to apply the parametrized configuration.

3.5.1 VDP Profile Instantiation (Detailed Troubleshooting)

With *feature evb* configured (either manually or due to 'Enable EVB Packet Trigger' being checked during POAP definitions), ECP packets (ethertype 0x8940) will be directed to the switch CPU to track the presence of VMs. VDP request packets are sent by default towards the leaf switch to the IEEE Nearest Bridge Destination Mac Address, 0180.c200.0000. When Nexus 1000V switches are behind UCS Fabric Interconnects, this destination MAC address will be terminated at Fabric Interconnects. In such a situation, the user can define an additional MAC address to receive VDP packets on, either in POAP or manually with **evb mac** *address* command.

```
acorn3# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
acorn3(config)# evb mac 0100.000d.0f0a

acorn3(config)# show evb

   EVB (Edge Virtual Bridge)

Role                            : VDP bridge
VDP MAC address                 : 0180.c200.0000 (Nearest Bridge)
                                  0100.000d.0f0a (User)
Resource wait init              : 25 (~ 335 sec)
Keep-alive init                 : 27 (~ 1342 sec)
No. received vdpdu              : 50472
No. dropped  vdpdu              : 0
No. received tlv                : 250850
No. received mgr tlv            : 50472
No. received assoc tlv          : 200378
No. received cmd                : 97216
```

Similarly, in this situation the Nexus 1000V needs to be instructed to send VDP packets on the non-default MAC address with the same configuration command:

```
dfa-n1kv1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
dfa-n1kv1(config)# evb mac 0100.000d.0f0a
dfa-n1kv1# show evb
                Edge Virtual Bridging
Role                             : VDP Station
VDP Mac Address                  : 0100.000D.0F0A
VDP Resource Wait Delay          : 20(17 secs)
VDP Reinit Keep Alive            : 20(10 secs)
```

The VDP protocol uses Edge Control Protocol (ECP) as a lower layer protocol to send its PDUs. ECP makes uses of sequence numbers and timer-based retransmission to provide a reliable delivery. ECP packets are acknowledged as well. If a given sequence number is not acknowledged for the duration of the retransmission timer (by default 163 milliseconds), the sending VDP Station or Bridge will retry three times by default before dropping the PDU and notifying the upper layer. The retransmission timer and the number of retries can be modified respectively with `**ecp retransmission-timer-exponent**' and '**ecp max-retries**` configuration commands on Nexus 1000V and leaf switches. The following command can be used on the DFA leaf switch to verify the operation and health of the ECP protocol.

```
acorn3# show ecp detail

   ECP (Edge Control Protocol)

Retrans timer init             : 14
Max retries                    : 3
Mode                           : LAN
No. rx packet                  : 265272
No. tx packet                  : 281847

1 Plugin(s):
ULP ID    Description          Status
--------- -------------------- --------
1         VDP Plugin           Enabled


1 Session(s):
Interface       S-Vlan  Peer MAC       Sess ID      RxSeq   TxSeq
--------------- ------- -------------- ------------ ------- -------
Eth1/26              1  0002.3d40.6403 1             36414   33264

[1]
Interface:     Ethernet1/26
Index:         0x1a019000
S-Vlan:        1
Sess ID:       1
Peer addr:     0002.3d40.6403
Rx seq:        36414
Tx seq:        33264
Rx
  141840 packets  12760 duplicate  0 drop
Tx
  114592 packets  12655 retry  3922 error
```

In the above example, a VDP Station has been detected over interface Ethernet1/26. Rx and Tx error counters for this session indicate ECP packets did not flow reliably at some point:

- **RX duplicate:** This counter is incremented when a PDU with a lower sequence number than the last received sequence number is received.
- **RX drop:** this counter is incremented when the receive queue size is full. As each packet should be acknowledged before the next packet is sent out, this should be an unusual situation.
- **Tx retry:** this counter is incremented when the last transmitted packet has not been acknowledged for the duration of the retransmission timer and a retry is attempted
- **Tx error:** this counter is increment when the last transmitted packet has not been acknowledged during the final retry.

From there, if the lower layer ECP was carrying a VDP packet, this packet will be sent to the evb process for further processing. The **show evb** command provides counters on the number of VDP PDUs and VDP TLVs received by the evb process:

```
acorn3# show evb

   EVB (Edge Virtual Bridge)

Role                           : VDP bridge
```

```
VDP MAC address                : 0180.c200.0000 (Nearest Bridge)
                                 0180.c200.0000 (User)
Resource wait init             : 25 (~ 335 sec)
Keep-alive init                : 27 (~ 1342 sec)
No. received vdpdu             : 3
No. dropped  vdpdu             : 0
No. received tlv               : 12
No. received mgr tlv           : 3
No. received assoc tlv         : 9
No. received cmd               : 0
```

In the below example, a VM connected to segment ID 31015 was associated, the dynamic VLAN 1604 was allocated for the virtual machine to send traffic is this segment, and the profile instantiation was successful:

```
acorn3# show evb hosts

EVB Host table

No. of Hosts: 1
No. of VSIs:  1

Flags: + - Multiple addresses
       > - Cisco OUI L3 address

Host Name       VNI      Vlan  BD    Mac-address    IP-Address        Interface
--------------- -------- ----- ----- -------------- ----------------- ---------------
TestVM1         31015    1604  1604  0050.56b9.72c6 10.1.15.200       Eth1/26


acorn3# show evb hosts detail

EVB Host table

No. of Hosts: 1
No. of VSIs:  1

Flags: + - Multiple addresses
       > - Cisco OUI L3 address

Host Name       VNI      Vlan  BD    Mac-address    IP-Address        Interface
--------------- -------- ----- ----- -------------- ----------------- ---------------
TestVM1         31015    1604  1604  0050.56b9.72c6 10.1.15.200       Eth1/26


Host Name:       TestVM1
Host UUID:       353020333920613320393420373020 36
  VSI ID:        00000000000000007927005056B972C6
    Interface:   Ethernet1/26
    Station:     0002.3d40.6403
    VNI:         31015
    VLAN:        1604
    BD:          1604
    MAC:         0050.56b9.72c6
    IP:          10.1.15.200
```

show evb vsi command will give additional information on a per virtual station interface basis. The output below shows a successful VSI association.

```
acorn3# show evb vsi detail

VSI entry table

No. of VSI entries: 1
No. of associated:  1

E-state: P  - Pre-associated
         PR - Pre-associated with reservation
         A  - Associated
         D  - De-associated
         S  - Standby

Mgr ID   VSI ID           Interface       Profile ID    M-state  E-State RWD/RKA
-------- ---------------- --------------- ------------- -------- ------- -------
00000000 7927005056B972C6 Eth1/26         00007927      WAIT_CMD       A    1336


MGR:           00000000000000000000000000000000
VSI ID:        00000000000000007927005056B972C6
Host name:     TestVM1
Interface:     Ethernet1/26
S-Channel:     1
Station:       0002.3d40.6403
Machine State: [08] VDP_BDG_WAIT_STATION_CMD
Entry State:   [03] VSI_ASSOCIATE
Keep-Alive:    1336 sec
Profile ID:    00007927
Filter[01]:    group=31015 vlan=1601 mac=0050.56b9.72c6
```

Below is a situation where the VSI could not be associated due to a missing profile for the segment in Cisco DCNM:

```
acorn3# show evb vsi detail

VSI entry table

No. of VSI entries: 1
No. of associated:  0

E-state: P  - Pre-associated
         PR - Pre-associated with reservation
         A  - Associated
         D  - De-associated
         S  - Standby

Mgr ID   VSI ID           Interface       Profile ID    M-state  E-State RWD/RKA
-------- ---------------- --------------- ------------- -------- ------- -------
00000000 7928005056B95942 Eth1/26                       INIT           -


MGR:           00000000000000000000000000000000
VSI ID:        00000000000000007928005056B95942
Host name:     TestVM2
Interface:     Ethernet1/26
S-Channel:     1
```

```
Station:       0002.3d40.6403
Machine State: [02] VDP_BDG_INIT
Entry State:   -
Reason:        Could not apply profile
Recycle in:    327 sec
```

In certain failure scenarios, a de-associate indication may be returned directly to the Nexus 1000V switch. In those situations, the VSI can be removed directly from the switch and it might not be possible to identify the reason with *show* commands during the association attempt window.

Upon a new VM association, The EVB process notifies the HMM process that a new host is present, and also notifies the host interface and the associated segment ID.

```
acorn3# show fabric forwarding internal event-history auto-config | i "VDP request"
2015 Nov  1 10:24:36.820927 hmm [4263]: [4268]: 31015:0000000000000000079270050056B972C6 Got VDP request
(trigger: 1 id:31015 mtype: 3) on Ethernet1/26
```

mtype:3 indicates an association request while *mtype:4* indicates a de-association request.

The following counters can be checked to see if the HMM process received VDP associations or de-associations.

```
acorn3# show fabric database host statistics
..
VDP Association Requests       11
VDP DeAssociation Requests      9
…
```

If the profile for the segment has already been instantiated, HMM will immediately return a success notification to the EVB process to proceed to a successful association.

In case of a new segment, HMM will add a host entry to its database, keyed by the VNI/segment ID for the network in case of segment-based VDP:

```
acorn3# show fabric database host
Active Host Entries
flags: L - Locally inserted, V - vPC+ inserted, R - Recovered, X - xlated Vlan
VNI      VLAN  STATE          FLAGS PROFILE(INSTANCE)
31015    1604  Profile Active  L     defaultNetworkUniversalEfProfile(instance_vni_31015_13)
```

The hmm process will then request an available VLAN from the pool of POAP defined dynamic VLANs (*system fabric dynamic-vlans*).

The leaf switch will then query the LDAP server to retrieve the attributes and attribute values associated with the profile that is to be applied. In case of a segment-based VDP trigger, the search key for an LDAP entry in the network database will be made of the segment ID advertised by the VDP station. If necessary, the switch will proceed with retrieving the tenant profile, the profile contents and will synchronize the profiles and fabric database host entry to the vPC peer.

3.6 VDP profile de-instantiation

The HMM process keeps track of VDP hosts on a per-segment basis. When all hosts have left a given segment, the leaf switch will un-apply the profile and free up the dynamically allocated resources.

VDP host deletion

When a VM is disconnected from the network, the VDP Station (Nexus 1000V) will signal the event by sending a VDP De-associate TLV to the VDP bridge (leaf switch). This will trigger the deletion of the VSI.

To account for situations where the Nexus 1000V switch cannot send de-association requests, such as in the case of unexpected resets, power outages or disconnection from the network, the leaf switch also maintains a countdown keepalive timer for each associated VSI. During the life of the VM association, the Nexus 1000V refreshes the state and the countdown keepalive timer on the leaf switch by periodically sending VDP Associate TLVs. These TLVs serve as keepalive messages, and are sent approximately every 10 seconds by default. Below you can verify the evb timers and VSI entries for a specific VNI.

```
acorn3# show evb

   EVB (Edge Virtual Bridge)

Role                            : VDP bridge
VDP MAC address                 : 0180.c200.0000 (Nearest Bridge)
                                  0180.c200.0000 (User)
Resource wait init              : 25 (~ 335 sec)
Keep-alive init                 : 27 (~ 1342 sec)
No. received vdpdu              : 1638
No. dropped  vdpdu              : 0
No. received tlv                : 6867
No. received mgr tlv            : 1638
No. received assoc tlv          : 5229
No. received cmd                : 10

acorn3# show evb vsi detail vni 31015

VSI entry table

No. of VSI entries: 1
No. of associated:  1

E-state: P  - Pre-associated
         PR - Pre-associated with reservation
         A  - Associated
         D  - De-associated
         S  - Standby

Mgr ID   VSI ID           Interface       Profile ID    M-state  E-State RWD/RKA
-------- ---------------- --------------- ------------- -------- ------- -------
00000000 7927005056B972C6 Eth1/26         00007927      WAIT_CMD       A    1339


MGR:           00000000000000000000000000000000
VSI ID:        00000000000000007927005056B972C6
Host name:     TestVM1
Interface:     Ethernet1/26
S-Channel:     1
```

```
Station:       0002.3d40.6403
Machine State: [08] VDP_BDG_WAIT_STATION_CMD
Entry State:   [03] VSI_ASSOCIATE
Keep-Alive:    1339 sec
Profile ID:    00007927
Filter[01]:    group=31015 vlan=1601 mac=0050.56b9.72c6
```

The keepalive countdown timer on the DFA leaf switch, which is by default approximately 22 minutes, can be modified using the **evb reinit-keep-alive** command:

```
acorn3(config)# evb reinit-keep-alive ?
  <22-31>  Timer exponent. (Min 22 exp ~ 40 seconds)
```

3.6.1 Profile deletion

The HMM process keeps track of the VDP hosts attached to a segment. In the below example, two VDP hosts are attached to the segment 31015:

```
acorn3# show fabric database host detail
Active Host Entries
flags: L - Locally inserted, V - vPC+ inserted, R - Recovered, X - xlated Vlan
VNI      VLAN  STATE           FLAGS PROFILE(INSTANCE)
31015    1601  Profile Active  L     defaultNetworkUniversalEfProfile(instance_vni_31015_15)
Displaying VDP hosts
Interface      Encap      Flags State           VSI-ID
Eth1/26        1601       L     Profile Active  00000000000000007927005056B95942
Eth1/26        1601       L     Profile Active  00000000000000007927005056B972C6
```

```
acorn3# show fabric database host vni 31015
instance_index 15
Got Local originated vdp type trigger at 15:36:44
Number of VDP Hosts: 2
Sent to Database Manager at 14:48:50
Received Parameters from Database Manager at 14:48:50
Displaying parameters for profile defaultNetworkUniversalEfProfile and instance instance_vni_31015_15
parameter 0: $gatewayIpAddress=10.1.15.1
parameter 1: $netMaskLength=24
parameter 2: $vlanId=
parameter 3: $segmentId=31015
parameter 4: $vrfName=PineForest:Tree1
parameter 5: $gatewayIpAddress=10.1.15.1
parameter 6: $netMaskLength=24
parameter 7: $dhcpServerAddr=5.0.3.91
parameter 8: $vrfDhcp=management
parameter 9: $gatewayIpv6Address=
parameter 10: $prefixLength=
parameter 11: $mtuValue=
parameter 12: $include_vrfSegmentId=50010
parameter 13: $segmentId=31015
parameter 14: $vlanId=1601
parameter 15: $asn=60100
Got VLAN allocated from vlan manager at 14:48:50
Sent Apply to Configuration Manager at 14:48:50
Completed executing all commands at 14:48:51
Displaying VDP hosts
```

```
Interface      Encap      Flags State           VSI-ID
Eth1/26        1601       L     Profile Active  00000000000000007927005056B95942
Eth1/26        1601       L     Profile Active  00000000000000007927005056B972C6
```

When the number of VDP hosts for the segment reaches 0, HMM will schedule the profile for deletion and place it in *Delete Holddown* state. The configuration deletion will occur after the cleanup interval has expired. This timer is by default 15 minutes, and is configurable using the **fabric database timer cleanup** command. If new VDP association requests are received during this cleanup interval for the given segment, the profile state will go back to an *active* state and will not be deleted.

```
acorn3# show fabric database host
Active Host Entries
flags: L - Locally inserted, V - vPC+ inserted, R - Recovered, X - xlated Vlan
VNI      VLAN  STATE           FLAGS PROFILE(INSTANCE)
31015    1601  Delete Holddown L     defaultNetworkUniversalEfProfile(instance_vni_31015_16)
acorn3# show fabric database host vni 31015
instance_index 16
Got Local originated vdp type trigger at 15:58:12
Number of VDP Hosts: 0
Profile will be un-applied in 880 seconds
Sent to Database Manager at 15:58:12
Received Parameters from Database Manager at 15:58:12
Displaying parameters for profile defaultNetworkUniversalEfProfile and instance instance_vni_31015_16
parameter 0: $gatewayIpAddress=10.1.15.1
parameter 1: $netMaskLength=24
parameter 2: $vlanId=
parameter 3: $segmentId=31015
parameter 4: $vrfName=PineForest:Tree1
parameter 5: $gatewayIpAddress=10.1.15.1
parameter 6: $netMaskLength=24
parameter 7: $dhcpServerAddr=5.0.3.91
parameter 8: $vrfDhcp=management
parameter 9: $gatewayIpv6Address=
parameter 10: $prefixLength=
parameter 11: $mtuValue=
parameter 12: $include_vrfSegmentId=50010
parameter 13: $segmentId=31015
parameter 14: $vlanId=1601
parameter 15: $asn=60100
Got VLAN allocated from vlan manager at 15:58:12
Sent Apply to Configuration Manager at 15:58:12
Completed executing all commands at 15:58:12
Got no hosts for this profile
```

After the cleanup interval has expired, the DFA leaf switch will not accept VDP requests for that segment for 2 minutes. This interval allows the system to perform the necessary configuration deletion before a profile for the same segment needs to instantiated again, avoiding a concurrent de-configuration and configuration. This timer can be changed with **fabric database timer vdp** configuration command.

```
acorn3# show fabric database host vni 31015
instance_index 0
Got Unknown originated vdp type trigger at 16:45:19
Number of VDP Hosts: 0
New VDP requests will be accepted in 100 seconds
Sent to Database Manager at 15:58:12
```

```
Received Parameters from Database Manager at 15:58:12
Got VLAN allocated from vlan manager at 15:58:12
Sent Apply to Configuration Manager at 15:58:12
Completed executing all commands at 15:58:12
Sent Un-apply to Configuration Manager at 16:51:43
Completed unapplying all commands at 16:51:44
Got no hosts for this profile
```

3.7 vPC+ and Auto-Profile Instantiation

Leaf switch vPC+ pairs synchronize their auto-profile and host learning states via Cisco Fabric Services (CFS). HMM communicates with the CFS process. CFS allows for leaf switches to synchronize their states via CFS messaging across the peer-link. Note that CFS is not a DFA specific feature however. With the below commands, you can verify that HMM is communicating with CFS, and that CFS is synchronizing remote hosts learned via HMM (event-history command).

```
dfa-n6k-leaf-1# show cfs internal application name hmm

Name: hmm, sap 1248
------------------------------
Enabled      : Yes
Scope        : Physical-eth
Timeout      : 30
Merge detail : Capable No, Version 0
Counters     : Uncoordinated 0, Unrestricted 0
App-id       : 1248 (0x4e0)
Region       : Default
IOD Enabled  : No
Lock details :
Lock Taken   : No, Lock Pending: 0

dfa-n6k-leaf-1# show fabric forwarding internal event-history trace  | grep vPC

2015 Nov 11 21:08:55.761690 hmm [4660]: [4662]: Processing vPC ack: reclaiming buffer 0x0x94fd2f4, xid:
0x12, count 1
2015 Nov 11 21:08:55.761248 hmm [4660]: [4662]: vPC: Received an ACK for type: vpc-sync-remote-hosts,
xid 0x12, ret_val: Success  from remote peer
2015 Nov 11 21:08:55.661719 hmm [4660]: [4665]: vPC thread [38] sending 1 vpc updates.
2015 Nov 11 21:08:55.661683 hmm [4660]: [4665]: Filled CFS header msg:vpc-sync-remote-hosts, ver:2,
xid:0x0, size:21 in vPC buffer
```

3.8 IPv6 and Auto-Profile Instantiation

IPv6 hosts can use the same HMM learning and auto-configuration process to install /128 entries in the iBGP routing table. This allows for IPv6 reachability across the DFA fabric.

```
dfa-n6k-leaf-1# show fabric forwarding ipv6 local-host-db vrf seinfeld:vandelay

HMM host IPv6 routing table information for VRF seinfeld:vandelay
Status: *-valid, x-deleted, c-cleaned in 00:09:40

    Host                MAC Address        SVI        Flags      Physical Interface
*   2000:aaaa::a/128    0000.0000.000a     Vlan100    0x20201    port-channel10
```

```
dfa-n6k-leaf-1# show ipv6 route vrf seinfeld:vandelay

IPv6 Routing Table for VRF "seinfeld:vandelay"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]

2000:aaaa::/64, ubest/mbest: 1/0, attached
    *via 2000:aaaa::1, Vlan100, [0/0], 2d00h, direct, , tag 12345
2000:aaaa::1/128, ubest/mbest: 1/0, attached
    *via 2000:aaaa::1, Vlan100, [0/0], 2d00h, local
2000:aaaa::a/128, ubest/mbest: 1/0, attached
    *via 2000:aaaa::a, Vlan100, [190/0], 1d23h, hmm
2000:aaaa::c/128, ubest/mbest: 1/0
    *via ::ffff:10.1.10.123%default:IPv4, [200/0], 04:37:29, bgp-65000, internal, tag 65000 segid 50000
```

3.9 Troubleshooting HMM Auto-Configuration Profile

Step 1: First check if the profile-map is setup properly.

Step 2: Check if the profiles are configured properly.

Step 2: Check Host status.

Step 4: Check HMM statistics.

Step 5: Check Syslogs for HMM errors.

Step 6: Look at the timestamps to see when the different tasks were completed.

4 Unicast Forwarding

Unicast Forwarding

DFA with a FabricPath underlay network relies on VN-Segment aware FabricPath encapsulation for sending frames through the fabric. The outer FabricPath destination is determined by the ingress leaf and depends on the forwarding mode that the ingress VLAN is configured for. In DFA, there are two modes that define the way packets are forwarded: Traditional Forwarding, also known as Anycast Gateway and Enhanced Forwarding, also known as Proxy Gateway. In both forwarding modes, the SVI is instantiated on every leaf switch that has a resource connected to it using the network. This allows VM Mobility across the fabric without the Gateway MAC or IP changing.

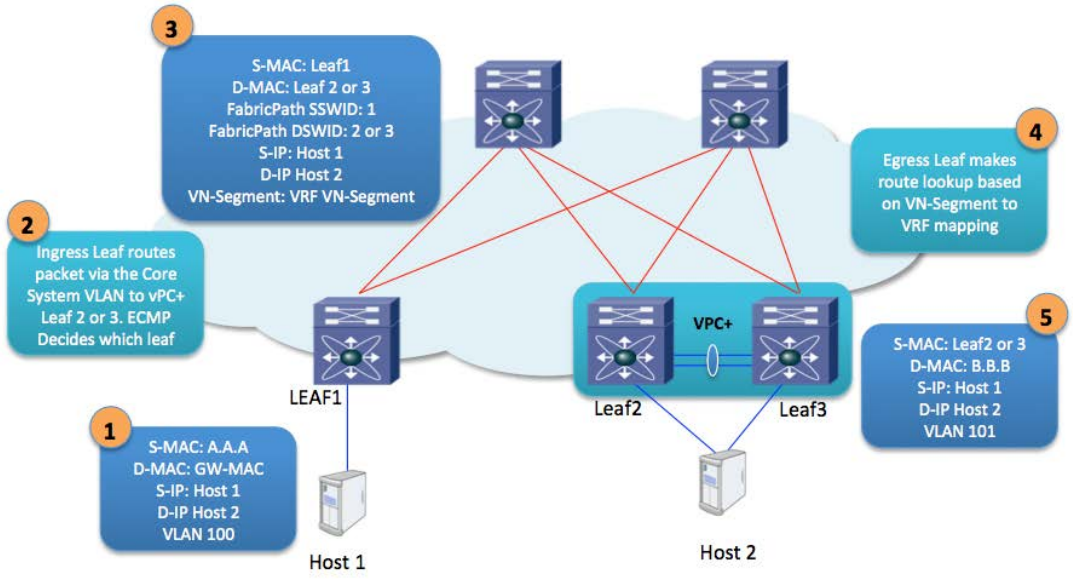
Enhanced Forwarding (Proxy Gateway)

In this mode, IP traffic from a connected host to another host is always Layer-3 routed irrespective of whether the destination is in the same subnet or different subnet as the source. To accomplish this, Proxy-Gateway will use a unique VN-Segment ID per tenant VRF. This VN-Segment is mapped to each tenant VRF using the System Dynamic

Core VLANs on each leaf switch. The VN-Segment mapping to Core System VLAN ID is locally significant but the VN-Segment mapping for this tenant is globally significant and needs to be consistent across all Leaf Switches.

Non IP traffic follows traditional fabricpath based forwarding.

Note: *The Core System VLAN is synonymous with the VXLAN term "Layer 3 VNI"*

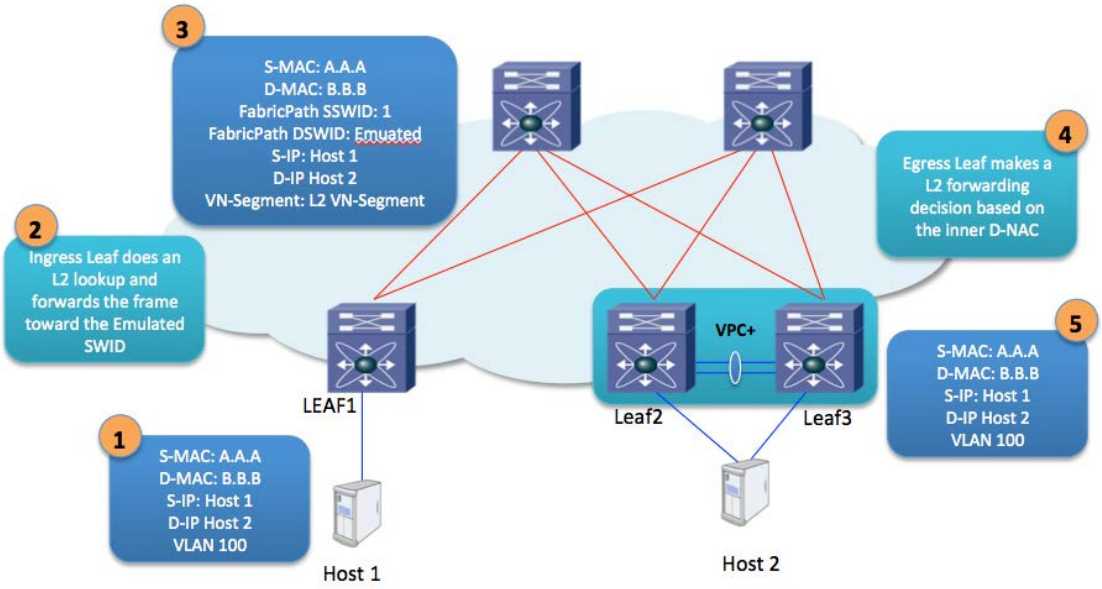


Traditional Forwarding (Anycast Gateway)

On a leaf switch, traffic from one host to another to another host within the same subnet is always Layer-2 bridged using the Layer-2 VN-Segment. The VN-Segment is again globally significant, but the VN-Segment to VLAN mapping on each leaf is locally significant to each mobility domain. Traffic that is routed between two VLANs enabled for Traditional Forwarding is routed using the Core System VLAN.

Non IP traffic follows traditional fabricpath based forwarding.

Note: *The Layer 2 VN-Segment used in Traditional Forwarding is synonymous with the Layer 2 VNI in VXLAN*



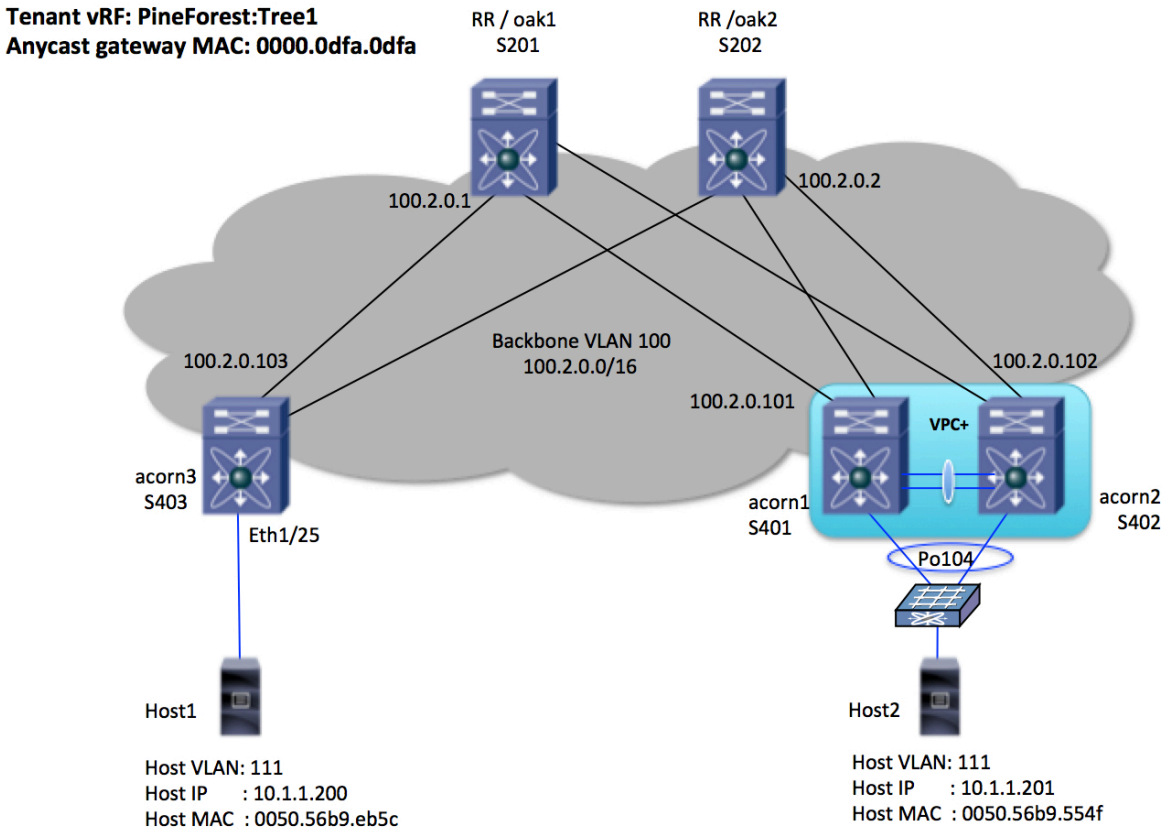
In both forwarding modes, leaf switches rely on having up to date routing information to make forwarding decisions. DFA uses BGP as its routing protocol of choice to propagate this information between leaf switches. To limit the number of BGP peers in a large-scale fabric, each leaf switch only peers with the BGP Route Reflector. The BGP Route Reflectors are placed on the spine switches. To import host routes into BGP, DFA uses a process called Host Mobility Manager, or HMM.

The table below outlines the key differences between Enhanced forwarding mode and Traditional Forwarding mode.

Key differences between Enhanced forwarding mode and Traditional Forwarding mode

	Enhanced Forwarding (Proxy-Gateway) Mode	Traditional Forwarding (Anycast-Gateway) Mode
ARP/IPv6 ND flooding in VLAN	No	Yes
Local Subnet Forwarding	Routed via System VLAN	Bridged
Across Subnet Forwarding	Routed via System VLAN	Routed via System VLAN
Unknown Unicast	Dropped	Flooded
Non IP traffic	Bridged using Fabricpath forwarding	Bridged using Fabricpath forwarding
Silent host Discovery	Cannot be Discovered	Can be Discovered

The following section highlights the forwarding differences in terms of ARP handling between Enhanced Forwarding and Traditional Forwarding for hosts in the same subnet.



Enhanced Forwarding

In the above topology, when Host 1 ARPs for Host 2, acorn3 will redirect the ARP request to the CPU and not flood it to the fabric. If the routing table has a route to Host 2, the ARP process will send an ARP response with the target MAC for Host 2 pointing to the Anycast Gateway MAC instead of Host 2's physical MAC address.

```
Host1@box:~$ arp -a
? (10.1.1.201) at 00:00:0d:fa:0d:fa [ether] on eth0
```

When host 1 sends a packet toward Host 2, a routing decision will be made on Host 1's leaf switch. Confirm that the VN-Segment mapped to the respective tenant VRF is indicated in the routing table as shown below.

```
acorn3# show ip route 10.1.1.201/32 vrf PineForest:Tree1
IP Route Table for VRF "PineForest:Tree1"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

10.1.1.201/32, ubest/mbest: 2/0
    *via 100.2.0.101%default, [200/0], 00:03:00, bgp-60100, internal, tag 60100, segid 50010
    *via 100.2.0.102%default, [200/0], 00:03:00, bgp-60100, internal, tag 60100, segid 50010
```

Traditional Forwarding

In Traditional forwarding the key difference is in how ARP will complete for hosts in the same subnet. When Host 1 sends an ARP request for Host 2, it is flooded through the fabric using a FabricPath FTAG. Host 2 will respond to the ARP requestsend an ARP response with its physical MAC address set as the target MAC address.

```
 Host1@box:~$ arp -a
? (10.1.1.201) at 00:50:56:b9:55:4f [ether] on eth0
```

When a packet from Host 1 to Host 2 arrives on Acorn3 destined to 0050.56b9.554f, a traditional MAC lookup will occur. The mac address will be done on the expected CE vlan (in this case vlan 111), which then needs to be confirmed to map to the expected VN-Segment

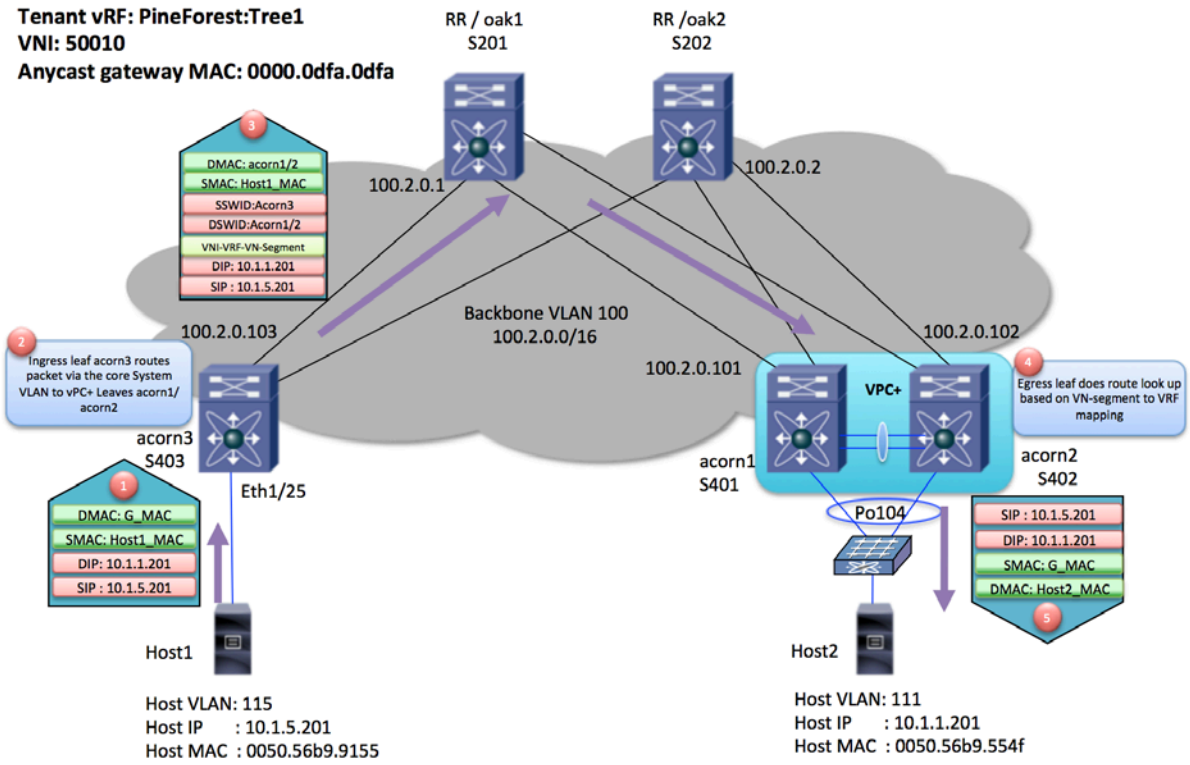
```
acorn3# show mac address-table address 0050.56b9.554f
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since last seen,+ - primary entry using vPC Peer-Link
   VLAN     MAC Address      Type      age     Secure NTFY   Ports/SWID.SSID.LID
---------+-----------------+--------+---------+------+----+------------------
* 111      0050.56b9.554f    dynamic   60         F    F  501.0.0

acorn3# show vlan id 111 vn-segment

VLAN Segment-id
---- -----------
111  31001
```

Unicast Packet Flow Example

In this example, a unicast packet from host1 in VLAN 115 attached to leaf acorn3, which is communicating with a vPC+ attached host2 in VLAN 111 to vPC+ leafs acorn1 and acorn2 will be traced. The methodology below applies to both traditional as well as enhanced forwarding modes.

1) Verifying IP connectivity in the Backbone VLAN between Leaf switches
2) Verifying the host route is installed into the BGP Table
3) Verifying forwarding path information

Acorn1, Acorn2 and Acorn3 are Cisco Nexus 6000 Series leaf switches. Oak1 and Oak2 are Cisco Nexus 7000 Series spine switches.



Verifying IP connectivity in the backbone VLAN

The backbone VLAN is defined on every switch through Cisco DCNM POAP process. This VLAN is used for BGP communication between all leaf switches and route reflectors. To limit ARP traffic in the fabric, FabricPath ISIS is used to share IP to MAC mappings for all leaf switches.

With the below command, you can verify the forwarding mode of the ingress Vlan.

```
acorn3# show run interface vlan 115 expand-port-profile

!Command: show running-config interface Vlan115 expand-port-profile
!Time: Thu Nov 12 15:50:03 2015

version 7.1(2)N1(1)

interface Vlan115
  no shutdown
  vrf member PineForest:Tree1
  no ip redirects
  ip address 10.1.5.1/24 tag 12345
  no ipv6 redirects
  fabric forwarding mode proxy-gateway
```

The tenant VLAN is configured on the switch using the **fabric forwarding control-segment**. The adjacencies to the peer switches is now shared via FabricPath ISIS. In the below output, the IP address of every leaf and spine switch which is part of the fabric will be accounted for.

```
acorn3# show ip adjacency

Flags: # - Adjacencies Throttled for Glean
       G - Adjacencies of vPC peer with G/W bit

IP Adjacency Table for VRF default
Total number of entries: 7
Address         MAC Address     Pref Source     Interface
100.2.0.1       f866.f203.2bc2  1    ISIS       Vlan100
100.2.0.2       0024.986f.4742  1    ISIS       Vlan100
100.2.0.3       b414.89e0.bdc6  1    ISIS       Vlan100
100.2.0.101     002a.6ab6.797c  1    ISIS       Vlan100
100.2.0.102     002a.6a5b.71fc  1    ISIS       Vlan100
100.2.0.105     002a.6a7e.c67c  1    ISIS       Vlan100
100.2.0.106     002a.6a7e.c37c  1    ISIS       Vlan100
```

To see what a local switch is advertising check the FabricPath ISIS information with the following command:

```
acorn3# show fabricpath isis protocol | inc Control
  System ID : 002a.6a66.cf7c  IS-Type : L1 Fabric-Control SVI: Vlan100
    Fabric Control MAC/IP/IPv6 address: 002a.6a66.cf7c/100.2.0.103/0
```

If a remote Adjacency is missing from the Adjacency table, then review the ISIS database via the following command

```
acorn3# show fabricpath isis database detail | inc "IP to MAC Mapping" next 4
      IP to MAC Mapping :
       MAC: 0024.986f.4742, # IPv4 addr 1, # IPv6 addr 0
         IPv4 address: 100.2.0.2
      Nickname Migration :
       Swid: 202 Sec. Swid: 0
```

Due to ISIS sharing Adjacency information, ARP resolution is not required for a remote Leaf and IP traffic can be forwarded between them. The below output depcits an empty ARP table, but still a successful ping.

```
acorn3# show ip arp 100.2.0.1

Flags: * - Adjacencies learnt on non-active FHRP router
       + - Adjacencies synced via CFSoE
       # - Adjacencies Throttled for Glean
       D - Static Adjacencies attached to down interface

IP ARP Table
Total number of entries: 0
Address         Age       MAC Address     Interface

acorn3# ping 100.2.0.1 count 1
PING 100.2.0.1 (100.2.0.1): 56 data bytes
64 bytes from 100.2.0.1: icmp_seq=0 ttl=254 time=1.283 ms

--- 100.2.0.1 ping statistics ---
1 packets transmitted, 1 packets received, 0.00% packet loss
round-trip min/avg/max = 1.283/1.282/1.283 ms
```

2) Verifying a host route is installed into BGP

When a host comes online in a DFA fabric, the switch relies on an ARP packet from the host to populate the HMM table, ARP and the adjacency manager. To ping Host 2, Host 1 needs to resolve ARP for the default Gateway. During ARP resolution, HMM will be populated on the leaf switch.

```
acorn3# show fabric forwarding ip local-host-db vrf PineForest:Tree1 10.1.5.201/32
HMM routing table information for VRF PineForest:Tree1, address family IPv4
HMM routing table entry for 10.1.5.201/32
Hosts: (1 available)

  Host type: Local(Flags: 0x20201), in Rib
  mac: 0050.56b9.9155, svi: Vlan115, bd: 115, phy_intf: Ethernet1/25
```

When viewing the HMM output it is important to verify the host MAC address as well as the interface the host is connected to. HMM will now update the URIB on the Leaf switch. The host route is learned via HMM instead of adjacency manager.

```
acorn3# show ip route 10.1.5.201/32 vrf PineForest:Tree1
IP Route Table for VRF "PineForest:Tree1"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

10.1.5.201/32, ubest/mbest: 1/0, attached
    *via 10.1.5.201, Vlan115, [190/0], 1d18h, hmm
```

After installing the /32 host route in URIB, an update will be sent to BGP. BGP will now advertise reachability for Host1 to the route reflectors through a VPNv4 update.

```
acorn3# show bgp vpnv4 unicast 10.1.5.201/32 vrf PineForest:Tree1
BGP routing table information for VRF default, address family VPNv4 Unicast
Route Distinguisher: 100.2.0.103:4    (VRF PineForest:Tree1)
BGP routing table entry for 10.1.5.201/32, version 265
Paths: (1 available, best #1)
Flags: (0x80c0502) on xmit-list, is not in urib, exported
  vpn: version 617, (0x100002) on xmit-list

  Advertised path-id 1, VPN AF advertised path-id 1
  Path type: redist, path is valid, is best path
  AS-Path: NONE, path locally originated
    0.0.0.0 (metric 0) from 0.0.0.0 (100.2.0.103)
      Origin incomplete, MED 0, localpref 100, weight 32768
      Extcommunity:
          RT:60100:50010
          SOO:0.0.0.0:0
          COST:pre-bestpath:96:1610612736

  VRF advertise information:
  Path-id 1 not advertised to any peer

  VPN AF advertise information:
  Path-id 1 advertised to peers:
    100.2.0.1          100.2.0.2
```

DFA Spine switches Oak 1 and 2 (.105 and .106), which are also the BGP route reflectors, will reflect this information in their BGP table:

```
oak1# show bgp vpnv4 unicast 10.1.5.201/32
BGP routing table information for VRF default, address family VPNv4 Unicast
Route Distinguisher: 100.2.0.103:4
BGP routing table entry for 10.1.5.201/32, version 54080
Paths: (2 available, best #2)
Flags: (0x000002) on xmit-list, is not in urib

  Advertised path-id 2
  Path type: internal, path is valid, not best reason: RR Cluster Length
  AS-Path: NONE, path sourced internal to AS
    100.2.0.103 (metric 0) from 100.2.0.2 (100.2.0.2)
      Origin incomplete, MED 0, localpref 100, weight 0
      Received label 524288
      Received path-id 1
      Extcommunity:
          RT:60100:50010
          SOO:0.0.0.0:0
          hex:03018060:60000000
          VNID:50010
      Originator: 100.2.0.103 Cluster list: 100.2.0.2

  Advertised path-id 1
  Path type: internal, path is valid, is best path
  AS-Path: NONE, path sourced internal to AS
    100.2.0.103 (metric 0) from 100.2.0.103 (100.2.0.103)
      Origin incomplete, MED 0, localpref 100, weight 0
      Received label 524288
```

```
      Extcommunity:
          RT:60100:50010
          SOO:0.0.0.0:0
          hex:03018060:60000000
          VNID:50010

  Path-id 1 advertised to peers:
    100.2.0.2          100.2.0.101        100.2.0.102        100.2.0.105
    100.2.0.106
  Path-id 2 advertised to peers:
    100.2.0.101        100.2.0.102        100.2.0.103        100.2.0.105
```

The route reflectors are responsible for sending these updates to all their BGP Peers, which ideally comprises of all leaf and spine switches in the fabric. If a route is missing between an RR and a leaf switch, focus on BGP troubleshooting (which is not specific to DFA).

3) Checking routing information for Host 1 and 2.

```
After resolving ARP for the default gateway, routes for Host 1 and 2 will be shared throughout the fab-
ric. Hardware forwarding will now occur for IP traffic between the hosts.
When host 1 sends a unicast frame to 10.1.1.201, Acorn3 will have its FIB programmed.
acorn3# show ip route 10.1.1.201 vrf PineForest:Tree1
IP Route Table for VRF "PineForest:Tree1"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

10.1.1.201/32, ubest/mbest: 2/0
    *via 100.2.0.101%default, [200/0], 4d19h, bgp-60100, internal, tag 60100, segid 50010
    *via 100.2.0.102%default, [200/0], 4d19h, bgp-60100, internal, tag 60100, segid 50010
```

The remote route is reachable via the core VLAN adjacency information in the default VRF. The route was learned through the BGP process with a route tag of 60100 and a route distinguisher of 60100:50010 (which equates to the AS number+Layer-3 VNI). When the switch forwards this frame though the FabricPath cloud, it will use the VN-Segment ID of 50010. When the traffic arrives at the remote switch, the VN-Segment will be associated with the correct tenant VRF.

To verify the vlan to VNI mapping use the following command.

```
acorn3# show vrf PineForest:Tree1 detail
VRF-Name: PineForest:Tree1, VRF-ID: 4, State: Up
    VPNID: unknown
    RD: 100.2.0.103:4
    VNI: 50010
    Max Routes: 0  Mid-Threshold: 0
    Table-ID: 0x80000003, AF: IPv6, Fwd-ID: 0x80000003, State: Up
    Table-ID: 0x00000003, AF: IPv4, Fwd-ID: 0x00000003, State: Up

acorn3# show vlan id 1-4000 vn-segment | inc 50010|VLAN
VLAN Segment-id
1501 50010
```

```
acorn3# show run | inc system | inc core
system fabric core-vlans 1501-1600

VLAN 1501 is system VLAN which was dynamically assigned to this VRF.
```

The switch will run an ECMP calculation to determine which switch to forward the traffic to in the vPC+ complex. Note that the physical FabricPath switch ID will be used and not the emulated FabricPath switch ID. The VN-Segment in which the traffic will be sent in is 50010.

For the next example we will use 100.2.0.102 as our next hop.

```
acorn3# show ip adjacency | inc 100.2.0.102
100.2.0.102     002a.6a5b.71fc  1    ISIS       Vlan100
```

The MAC address associated with the next hop is that of Acorn 2, or FabricPath Switch ID 402. To verify use the following command

```
acorn3# show mac address-table address 002a.6a5b.71fc vlan 100
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since last seen,+ - primary entry using vPC Peer-Link
   VLAN     MAC Address      Type      age     Secure NTFY   Ports/SWID.SSID.LID
---------+-----------------+--------+---------+------+----+------------------
* 100      002a.6a5b.71fc    static    0          F    F  402.0.0
```

To determine which path the packet would take though the FabricPath domain, the route can be viewed, as shown below.

```
acorn3# show fabricpath route switchid 402
FabricPath Unicast Route Table
'a/b/c' denotes ftag/switch-id/subswitch-id
'[x/y]' denotes [admin distance/metric]
ftag 0 is local ftag
subswitch-id 0 is default subswitch-id


FabricPath Unicast Route Table for Topology-Default

1/402/0, number of next-hops: 4
        via Eth2/1/1, [115/80], 4 day/s 20:30:22, isis_fabricpath-default
        via Eth2/1/2, [115/80], 4 day/s 20:30:22, isis_fabricpath-default
        via Eth2/1/3, [115/80], 4 day/s 20:30:22, isis_fabricpath-default
        via Eth2/1/4, [115/80], 4 day/s 20:30:22, isis_fabricpath-default
```

For verification, an ELAM capture has been used to trace the packet in hardware. More in-depth guides for ELAM captures on Cisco Nexus switches can be found on Cisco.com, and the guides include Cisco Nexus 7000, 6000, and,5600 Series switch guides. The output below is an egress ELAM from Nexus6000 leaf switch acorn3 and has been truncated to highlight important fields.

```
acorn3# elam slot all
```

```
acorn3(bigsur-elam)# trigger lu egress ipv4 if source-ipv4-address_ipv4 10.1.5.201 destination-ipv4-
address_ipv4 10.1.1.201
acorn3(bigsur-elam)# start capture
acorn3(bigsur-elam)# show capture lu
Egress Interface: Ethernet2/1/3 IS NOT A PC

acorn3(bigsur-elam)# show capture rs
+----------------------------------------+
|               Result Vector            |
+---------------------+------------------+
|       Field         |      Raw Value   |
+---------------------+------------------+
| CE_DA               | 0x002a6a5b71fc   |
| CE_SA               | 0x002a6a66cf7c   |
| CE_Q0_VLAN          | 12               |
| CE_Q1_VLAN          | 858              |
| L3_DA               | 10.1.1.201       |
| L3_SA               | 10.1.5.201       |
| CDCE_DA             | 0x020192000000   |
| CDCE_SA             | 0x020193000000   |
+---------------------+------------------+
```

CE_DA	The Classical Ethernet (CE) **D**estination **A**ddress was derived from our ECMP Next hop when checking the IPv4 route for Host 2.
CE_SA	The **CE S**ource **A**ddress is the local MAC address of Acorn 3
CE_Q0_VLAN CE_Q1_VLAN	VN-Segmentation is accomplished in hardware using a double dot1q tag. 12 and 858 have to be converted to a 3-byte value and concatenate the strings.
Example:	Q0: 12 = 0x00c Q1: 858 = 0x35a VN-Segment: 0x00c35a= 50010.
L3_DA	Destination IP address for Host2
L3_SA	Source IP address from Host1
CDCE_DA CDCE_SA	The CDCE, **C**isco **D**ata **C**enter **E**thernet refers to the FabricPath En-capsulation. A FabricPath MAC breakdown is out of scope of this document.
Example:	0x193 = Source Switch ID of 403 (Acorn 3) 0x192 = Destination Switch ID of 402 (Acorn 2)

The ELAM output confirms the software programming for the next hop Mac address and destination Switch ID. The packet will leave Acorn 3 via Eth2/1/3.

```
acorn3# show cdp neighbors interface ethernet 2/1/3
Capability Codes: R - Router, T - Trans-Bridge, B - Source-Route-Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater,
                  V - VoIP-Phone, D - Remotely-Managed-Device,
                  s - Supports-STP-Dispute


Device-ID           Local Intrfce   Hldtme Capability  Platform      Port ID
oak2(TBM13052934)   Eth2/1/3        141    R S I s     N7K-C7010     Eth7/5
```

Oak2 is a DFA spine switch. Spine switches are in FabricPath transit mode by default, and are not VN-Segment aware. When a FabricPath encapsulated frame comes in and the CE_Q0_VLAN is not defined on the switch, a special FabricPath scale VLAN will be used to make a forwarding decision. This is different than traditional FabricPath forwarding where the frame would have been discarded on ingress. To verify the usage of this FabricPath scale vlan, use the following command.

```
oak1# show vlan internal usage

VLANs                   DESCRIPTION
------------------      -----------------
3968-4031               Multicast
4032-4035,4048-4059     Online Diagnostic
4036-4039,4060-4087     ERSPAN
4042                    Satellite
4044                    Native VLAN to enable/disable tagging
4040                    Fabric scale
3968-4095               Current
```

The packet from Acorn3 was sent with a Destination Switch ID of 402. 402 can be reached via Eth7/4 from spine switch Oak1.

```
oak1# show fabricpath route switchid 402
FabricPath Unicast Route Table
'a/b/c' denotes ftag/switch-id/subswitch-id
'[x/y]' denotes [admin distance/metric]
ftag 0 is local ftag
subswitch-id 0 is default subswitch-id


FabricPath Unicast Route Table for Topology-Default

1/402/0, number of next-hops: 1
        via Eth7/4, [115/40], 4 day/s 20:50:32, isis_fabricpath-default

oak1# show cdp neighbors interface ethernet 7/4 | inc acron2 next 1
acorn2(FOC1752R0CG)
                    Eth7/4         153    R S I s   N6K-C6004     Eth1/1/2
```

The packet will be received on Acorn2 destined to the system MAC address resulting in a routing lookup. The Routing table is determined by the VN-Segment to VRF Tenant mapping shown below. Host2 related information corresponds to the PineForest:Tree1VRF for VLAN 111, on Po104.

```
acorn2# show vlan id 1-4000 vn-segment | inc VLAN|50010
VLAN Segment-id
1502 50010

acorn2# show vrf PineForest:Tree1 detail
VRF-Name: PineForest:Tree1, VRF-ID: 14, State: Up
    VPNID: unknown
    RD: 100.2.0.102:14
    VNI: 50010
    Max Routes: 0  Mid-Threshold: 0
    Table-ID: 0x8000000d, AF: IPv6, Fwd-ID: 0x8000000d, State: Up
    Table-ID: 0x0000000d, AF: IPv4, Fwd-ID: 0x0000000d, State: Up
```

The destination IP address will be checked against the PineForest:Tree1 VRF.

```
acorn2# show ip route 10.1.1.201 vrf PineForest:Tree1
IP Route Table for VRF "PineForest:Tree1"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

10.1.1.201/32, ubest/mbest: 1/0, attached
    *via 10.1.1.201, Vlan111, [190/0], 5d04h, hmm
```

Since the route points out a local interface we can confirm the correct next hop MAC address via ARP or HMM.

```
acorn2# show ip arp vrf PineForest:Tree1 | grep 10.1.1.201
10.1.1.201      00:00:02  0050.56b9.554f  Vlan111         +

acorn2# show mac address-table vlan 1601 | grep 554f
  VLAN/BD   MAC Address      Type      age     Secure NTFY Ports/SWID.SSID.LID
---------+-----------------+--------+---------+------+----+------------------
* 111     0050.56b9.554f    dynamic   1800       F    F     Po104

The member ports of Po104 are Eth1/4/3 and Eth1/4/4.

acorn2# show port-channel summary interface port-channel 104 | begin Group
--------------------------------------------------------------------------------
Group Port-       Type     Protocol  Member Ports
      Channel
--------------------------------------------------------------------------------
104   Po104(SU)   Eth      LACP      Eth1/4/3(P)  Eth1/4/4(P)
acorn2#
```

To confirm the packet forwarding in hardware an ELAM can be used to see which hardware path the packet will take. To confirm the egress VLAN an egress ELAM has to be used. A verification sample is given below.

```
acorn2# elam slot all
```

```
acorn2(bigsur-elam)# trigger lu ingress ipv4 if source-ipv4-address_ipv4 10.1.5.201 destination-ipv4-
address_ipv4 10.1.1.201
acorn2(bigsur-elam)# start capture
acorn2(bigsur-elam)# show capture lu
Ingress Interface: Ethernet1/1/2 IS NOT A PC
+----------------------------------------+
|              Lookup Vector             |
+--------------------+-------------------+
|       Field        |      Raw Value    |
+--------------------+-------------------+
| CDCE_DA            | 0x020192000000       |
| CDCE_SA            | 0x020193000000       |
| CE_DA              | 0x002a6a5b71fc       |
| CE_SA              | 0x002a6a66cf7c       |
| CE_Q0_VLAN         | 12                   |
| CE_Q1_VLAN         | 858                  |
| L3_SA              | 10.1.5.201           |
| L3_DA              | 10.1.1.201           |
+--------------------+-------------------+
acorn2(bigsur-elam)# trigger lu egress ipv4 if source-ipv4-address_ipv4 10.1.5.201 destination-ipv4-
address_ipv4 10.1.1.201
acorn2(bigsur-elam)# start capture
acorn2(bigsur-elam)# show capture lu
Egress Interface: Ethernet1/4/3 IS PC
acorn2(bigsur-elam)# show capture rs
+----------------------------------------+
|              Result Vector             |
+--------------------+-------------------+
|       Field        |      Raw Value    |
+--------------------+-------------------+
| CE_DA              | 0x005056b9554f      |
| CE_SA              | 0x002a6a5b71fc      |
| CE_Q0_VLAN         | 111                 |
| L3_DA              | 10.1.1.201          |
| L3_SA              | 10.1.5.201          |
| EXT_VLAN           | 111                 |
| CDCE_DA            | 0x0201f500000d      |
| CDCE_SA            | 0x020192000000      |
+--------------------+-------------------+
acorn2(bigsur-elam)#
```

5 Multicast Forwarding

We will cover both multicast control plane and data-plane forwarding. For each section we will start with a high-level functional overview, followed by detailed troubleshooting and verification CLI outputs. A DFA multicast cheat sheet will also be provided in the Cheat Sheets section.

5.1 Multicast Forwarding Control Plane

Within DFA topologies, we support PIM Any-Source Multicast (ASM), PIM Source-Specific Multicast (SSM), and PIM Bidirectional (Bidir) multicast forwarding. For the purposes of this document we will focus on ASM functioning, and will point out relevant information for SSM and Bidir as needed. For DFA with a Fabricpath underlay, IGMP is terminated at each respective leaf switch and is not propagated through the rest of the fabric, as is the case in non-DFA Fabricpath topologies. Instead, we utilize BGP Subsequent Address Family Identifiers (SAFI) updates to announce remote receiver

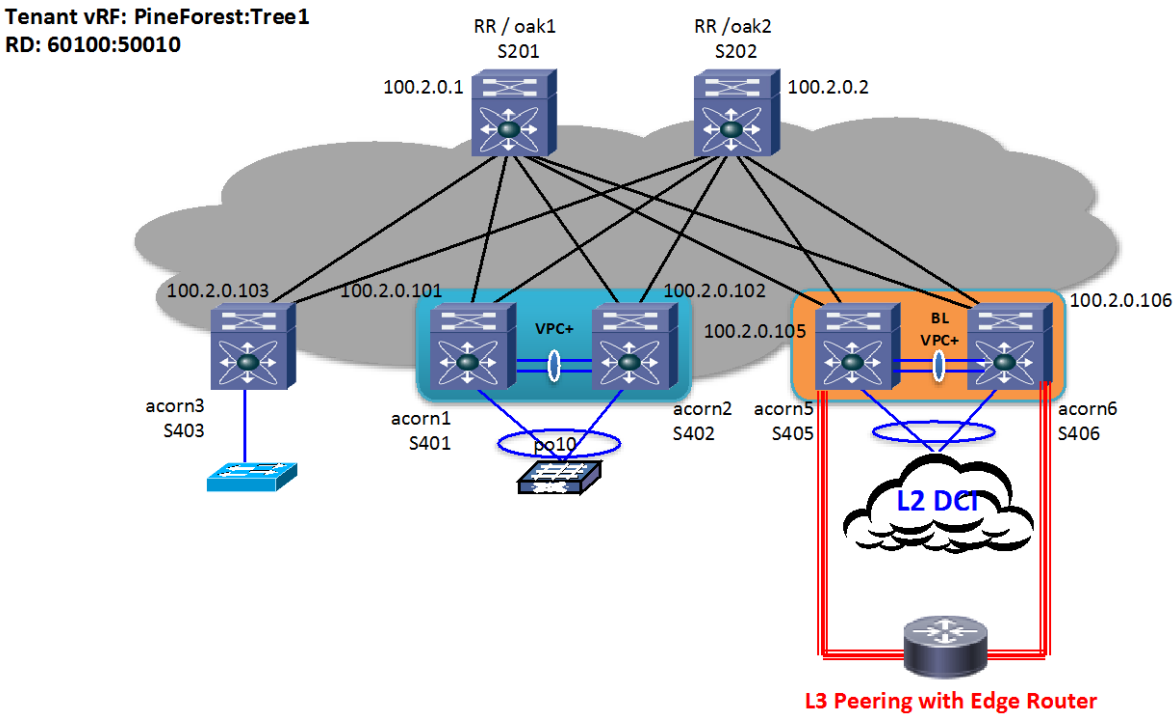
interest across the fabric. These BGP updates are sent across the backbone VLAN, in the default VRF, with a route distinguisher made up of AS number and the Layer-3 VNI for the respective tenant VRF. For example, in our topology below, the AS number is 60100 and the Layer-3 VNI for our tenant VRF is 50010, giving us a route distinguisher for this tenant VRF of 60100:50010.

By default, PIM functions in *PIM passive mode.* This means that SVIs (or in the case of N7k, BDIs) on server leaf switches will not have PIM explicitly configured, but PIM will still perform specific functions. SVIs operating in PIM passive mode will not send PIM packets (hellos, joins, or prunes), as these functions will be handled through BGP updates through the fabric instead. PIM will however perform First Hop Router (FHR) source registration with rendezvous points (RPs) just as it would in non-DFA PIM environments. RPs can be configured at border leaf switches or only outside of the fabric. Server Spine switches and server leaf switches are not supported as RPs.

We will utilize the topology below in each of the following sections. Tenant VRF will be PineForest:Tree1 with a route distinguisher of 60100:50010.

Note: All commands are run in the respective tenant VRF context (you can enter the respective VRF by using the **routing-context vrf** *name* command.

Base Topology



DFA Multicast Software Architecture and Tables

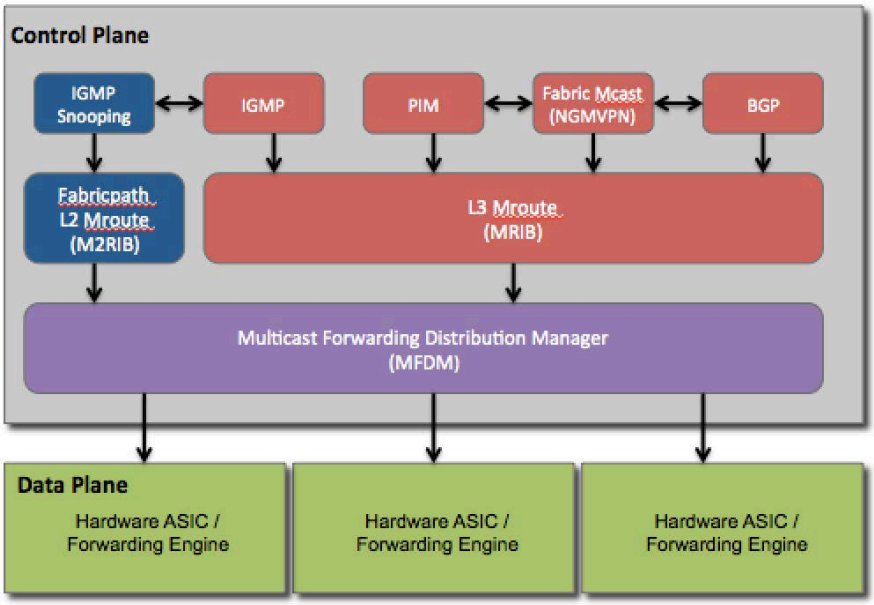


Table	CLI	Information
IGMP Snooping Table	show ip igmp snooping	Stores information on received joins for server facing vlans. Updates fabricpath mroute table
L2 Fabricpath Mroute Table	show fabricpath mroute	Stores L2 forwarding information for vni + group and the L2 oif provided by IGMP snooping.
L3 Mroute Table (MRIB)	show ip mroute	Standard mroute table storing L3 incoming interface and L3 outgoing interface.
Fabric Multicast Table	show fabric multicast ipv4 <mroute|rp-grange|ssm-range>	Stores SSM range, RP-Group mapping and information of interested leaf nodes for (*,G), and (S,G) information and the respective RPF neighbor selected for each entry.

5.1.1 Rendezvous Point Propagation

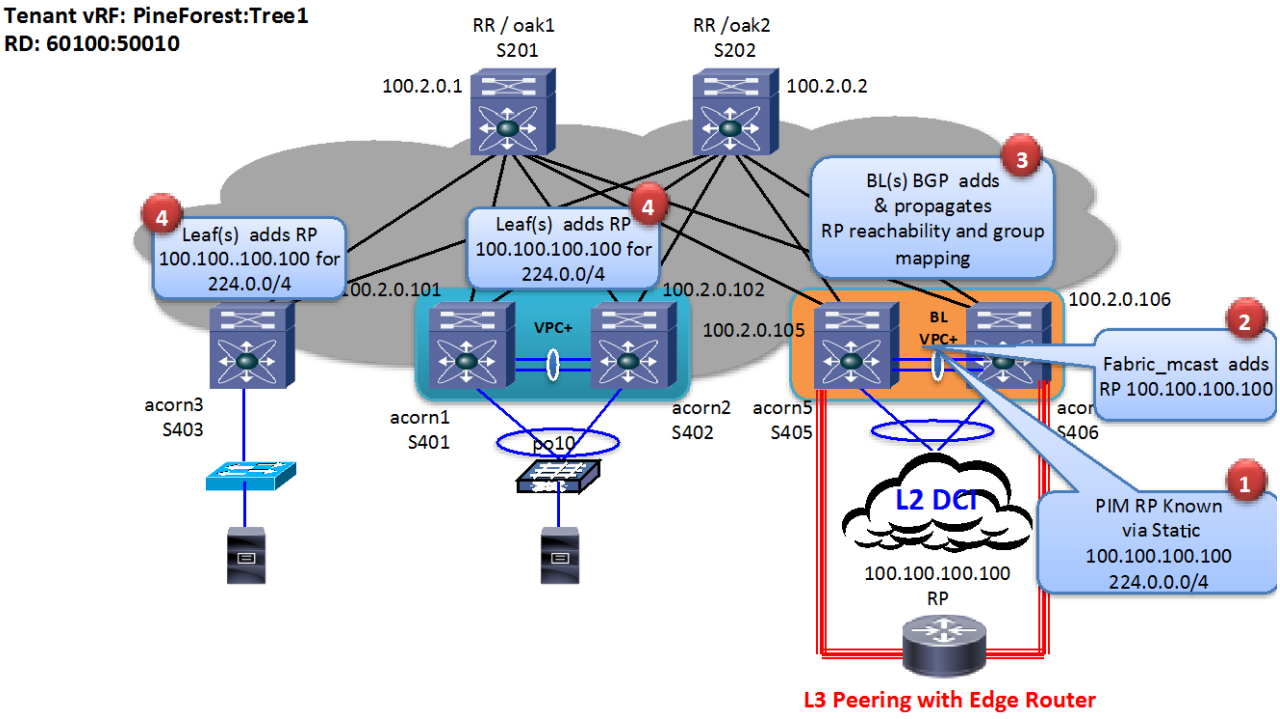
Rendezvous Points are supported either on border leaf switches or only outside the fabric. Server leaf switches or spine switches are not supported as RPs. If the RP is located outside the fabric, then the border leaf switches will learn of the RP-to-group mappings through one of the standard mechanisms of static configuration, Auto-RP, or BSR. Once the BL switches have been informed of the RP-to-group mappings, they will advertise this information into the fabric on the backbone VLAN via BGP SAFI messages. The BL switches will not use Auto-RP, BSR, or any PIM messaging to inform server leaf switches of the RP-to-group mappings. Redundant border leaf switches should have the same view of RP-to-group mappings, but if for some reason they do not, then the border leaf switch with higher backbone VLAN IP address will be preferred by server leaf switches in the fabric.

Note: *Non-default SSM group range (232.0.0.0/8) information is propagated by the border leaf switches in the same manner as RP-to-group range information. This is done in order to prevent having to configure this on each individual leaf.*

FHR source registration between FHR and RP is done via the same PIM unicast registration message process as it does in non-DFA environments. One notable difference is that since multiple server leaf switches will have the same gateway IP address used to send the PIM FHR registration message, administrators have the option to use the **ip pim register-source** *interface* command to specify a single, unique IP address on each server leaf switch to be used for source registration.

For this example, we will be verifying that acorn3 learns the RP 100.100.100.100, which is located outside of the fabric.

High-level Overview for RP Propagation



1. The PIM process on the border leaf switches will learn the RP-to-group mappings (RP 100.100.100.100 for multicast group 224.0.0.0/4) via static configuration.
2. PIM on the border leaf switches will then update the fabric multicast table with the mapping information so that it can be passed to BGP.
3. Once BGP learns the mapping information, it will send out an RP-to-group mapping BGP SAFI update to the RRs, which will disperse the update to all the server leaf switches.
4. Server leaf switches will add the RP-to-group mapping and RP reachability information to BGP, fabric multicast, and PIM tables.

Detailed Troubleshooting and Verification Steps for RP Propagation

1. Start by confirming that the border leaf switches have learned of the RP-to-Group mapping information. In our case here, we have statically configured the RP-to-Group mapping on both border leaf switches. We will only show the verification output for one of the border leaf switches, acorn5, as the outputs should be the same. Always verify both border leaf switches' output if there is suspicion of a problem. The RP is the edge router, 100.100.100.100, and is RP for all groups, 224.0.0.0/4.

```
acorn5%PineForest:Tree1# show ip pim rp
PIM RP Status Information for VRF "PineForest:Tree1"
BSR disabled
Auto-RP disabled
BSR RP Candidate policy: None
BSR RP policy: None
```

```
Auto-RP Announce policy: None
Auto-RP Discovery policy: None

RP: 100.100.100.100, (0), uptime: 05:16:38, expires: never,
  priority: 0, RP-source: (local), group ranges:
      224.0.0.0/4
```

2. Confirm that the PIM process has passed this information to the fabric multicast and BGP tables, so that a BGP SAFI update can be sent to the RRs. The NLRI Origin IP address for the border leaf switch will be 0.0.0.0 because the BL itself is the next hop for the RP-to-Group reachability for devices in the fabric i.e. the BL is the origin of the update.

```
acorn5%PineForest:Tree1# show fabric multicast ipv4 rp-grange
VRF "PineForest:Tree1" RP Grange Database VNI: 50010

Information about Border Leaf Node: 0.0.0.0
NLRI Origin: 0.0.0.0 RP: 100.100.100.100 Group Range: 224.0.0.0/4 Prot_source: 1 Static    Hash len:
0 Priority: 0
```

```
acorn5%PineForest:Tree1# show bgp ipv4 mvpn rd 60100:50010
BGP routing table information for VRF default, address family IPv4 MVPN
BGP table version is 482, local router ID is 100.2.0.105

   Network            Next Hop            Metric     LocPrf     Weight Path
Route Distinguisher: 60100:50010    (Group-to-RP)
*>l[100.100.100.100][224.0.0.0][4][0.0.0.0][4][0][0]/128
                      0.0.0.0                            100      32768 i
```

3. The information in the BGP table above will then be put into a BGP SAFI update and forwarded to the route reflectors Oak1 and Oak2. Previously, on the border leaf switch acorn5, the next-hop information was 0.0.0.0 because acorn5 was the originating router. Here on the route-reflector/spine-1, we see that the next hop now points to acorn5's IP address of 100.2.0.105 as well as the acorn6 IP address of 100.2.0.106 since both border leaf switches have the same RP-to-Group mapping information and both sent a BGP update. (Again, we will show the information from only one RR/spine switch. In a redundant RR/spine switch topology, both would have the same information)

```
Oak1# show bgp ipv4 mvpn rd 60100:50010
BGP routing table information for VRF default, address family IPv4 MVPN
BGP table version is 886, local router ID is 100.2.0.1

   Network            Next Hop            Metric     LocPrf     Weight Path Route
Route Distinguisher: 60100:50010    (Group-to-RP)
*>i[100.100.100.100][224.0.0.0][4][0.0.0.0][4][0][0]/128
                      100.2.0.105                        100          0 i
*>i                   100.2.0.106                        100          0 i
```

4. Both route reflectors will then forward this BGP update to all leaf switches, and those leaf switches should show the same information in their respective BGP tables. We will look at acorn3 for verification. Notice that acorn3 below will show two routes, which is because we have redundant route reflectors sending the same BGP update.

```
acorn3%PineForest:Tree1# show bgp ipv4 mvpn rd 60100:50010
BGP routing table information for VRF default, address family IPv4 MVPN
BGP table version is 503, local router ID is 100.2.0.103

   Network            Next Hop            Metric     LocPrf     Weight Path Route
Route Distinguisher: 60100:50010    (Group-to-RP)
*>i[100.100.100.100][224.0.0.0][4][0.0.0.0][4][0][0]/128
                      100.2.0.105                       100          0 i
* i                   100.2.0.105                       100          0 i
*>i                   100.2.0.106                       100          0 i
* i                   100.2.0.106                       100          0 i
```

5. Now that we have verified that the server leaf switch's BGP table has proper RP-to-Group mapping, confirm that this information is passed down to the fabric multicast and PIM tables, and that you have a path to the RP address of 100.100.100.100. This unicast path towards the RP will be an ECMP default route originated by both border leaf switches. These ECMP routes both have the tag of the AS (60100), and the respective VRF Segment ID (L3-VNI) of 50010

```
acorn3%PineForest:Tree1# show fabric multicast ipv4 rp-grange
VRF "PineForest:Tree1" RP Grange Database VNI: 50010

Information about Border Leaf Node: 100.2.0.105
NLRI Origin: 100.2.0.105 RP: 100.100.100.100 Group Range: 224.0.0.0/4 Prot_source: 1 Static    Hash
len: 0 Priority: 0
Information about Border Leaf Node: 100.2.0.106
NLRI Origin: 100.2.0.106 RP: 100.100.100.100 Group Range: 224.0.0.0/4 Prot_source: 1 Static    Hash
len: 0 Priority: 0

acorn3%PineForest:Tree1# show ip pim rp
PIM RP Status Information for VRF "PineForest:Tree1"

RP: 100.100.100.100, (0), uptime: 06:03:14, expires: never,
  priority: 0, RP-source: (local), group ranges:
      224.0.0.0/4

acorn3%PineForest:Tree1# show ip route 100.100.100.100
IP Route Table for VRF "PineForest:Tree1"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

0.0.0.0/0, ubest/mbest: 2/0
    *via 100.2.0.106%default, [200/0], 09:21:07, bgp-60100, internal, tag 60100, segid 50010
    *via 100.2.0.105%default, [200/0], 09:21:07, bgp-60100, internal, tag 60100, segid 50010
```

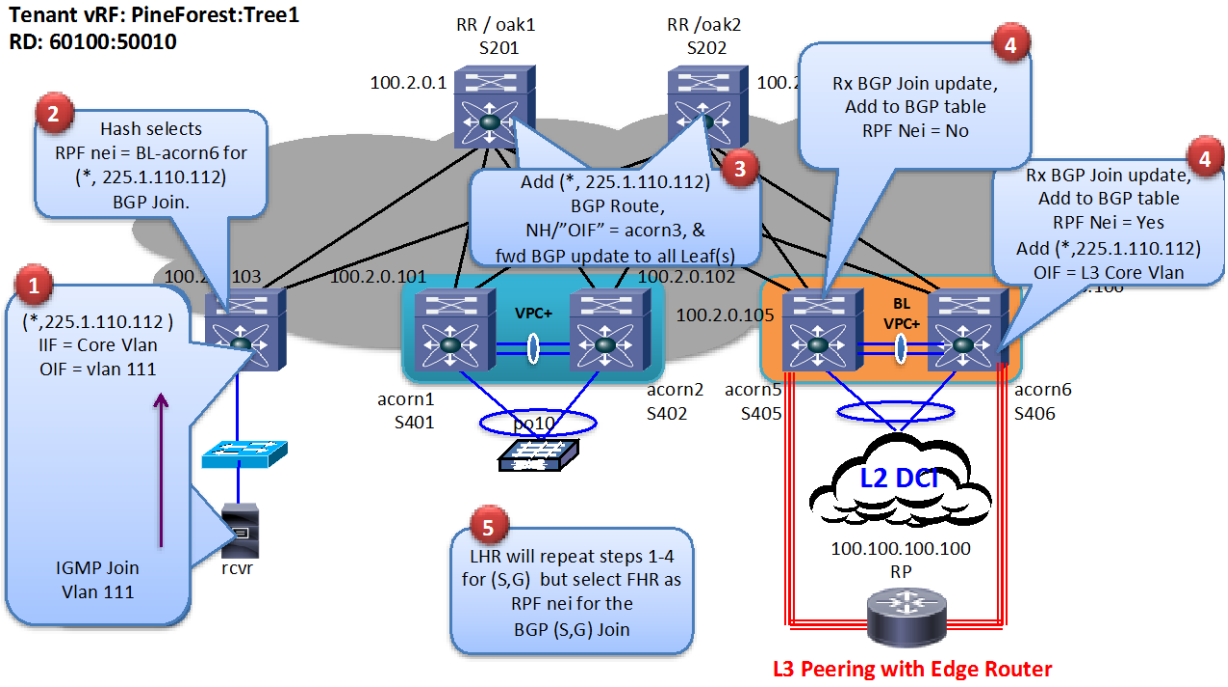
You have now successfully verified RP propagation through the fabric from the border leaf switches to the server leaf switches, and have the commands needed to isolate the device that is having a problem, if troubleshooting is needed.

5.1.2 BGP Join Propagation

Multicast receivers announce their interest for a group via IGMP. DFA with a Fabricpath underlay does not propagate IGMP information through the rest of the network via Group-Membership Link-State PDUs (GM-LSPs), nor does it forward the native IGMP packet onto the fabric (as is the case of non-DFA Fabricpath topologies). IGMP information is terminated at the receiving leaf and receiver interest is then translated into a BGP (*,G) or (S,G) join that will be transmitted through the fabric toward the RP or FHR respectively).

For this example we will be verifying, the (*,G) BGP Join propagation for 225.1.110.112, from the Last Hop Router (LHR) acorn3, through the fabric toward acorn5, and eventually destined to the Rendezvous Point. The RP is the same as in the previous example, 100.100.100.100, and is outside of the fabric.

High-level Overview for BGP Join Propagation



1. An IGMP join is sent in on the CE VLAN 111, and will create (*,G) state + multicast outgoing interface (OIF) at the LHR. The incoming interface (IIF) will be the Layer-3 core VLAN (VLAN 1502), and the OIF will be CE VLAN (VLAN 111)
    - The IIF is of the Layer-3 core VLAN because this is the path towards the border leaf switch(es) advertising RP reachability.
2. LHR acorn3 will then use a hash algorithm(*) to select a reverse path forwarding (RPF) neighbor that is announcing RP reachability to forward the BGP (*,G) Join message to the border leaf switches via the BGP route reflectors.

- The border leaf switch(es) should be the only device announcing reachability towards the RP, since the RP can only be configured on the border leaf switch or behind the border leaf switch outside of the fabric. Therefore, the hash would be used to select which BL is chosen.

3. The route reflector spine switches will receive this (*,G) BGP Join, update their BGP tables, and forward this update to all iBGP neighbors.
    - All leaf switches will add this update to their BGP and fabric multicast tables, but only the RPF neighbor (one of the border leaf switches) that was chosen by the hash algorithm on the LHR acorn3 will add the (*,G) entry into its mroute table (MRIB).
4. Once the border leaf switch receives this update and confirms itself to be the RPF neighbor selected, it will add a (*,G) + OIF into its mroute table (MRIB). The OIF will be the Layer-3 core VLAN (VLAN 1510) for the respective tenant VRF. IIF will be the DCI sub-interface towards the RP edge router (Po10.100).
    - The core VLAN on the border leaf switch (1510) and the core VLAN on the LHR leaf switch (1502) do not have to match VLAN IDs, but the key is that both server leaf and border leaf core VLANs map to the same VNI, 50010
5. This will cause the traffic to be forwarded via the (*,G) path to the LHR acorn3. Once LHR receives the multicast traffic, it will then begin the shortest path tree (SPT) cutover (just as in non-DFA environments), and trigger the BGP (S,G) Join. The above process will repeat with the BGP (S,G) Join destined to the FHR.

Note: The hash algorithm procedures are similar to the upstream-multicast hop (UMH) selection (per RFC 6513).

Detailed Troubleshooting and Verification Steps for BGP Join Propagation

LHR Verification

1. The first step is to ensure an IGMP Join was received from an interested receiver. IGMP snooping will program the interface the join was received on in the snooping table.

```
acorn3%PineForest:Tree1# show ip igmp snooping groups vlan 111
Type: S - Static, D - Dynamic, R - Router port, F - Fabricpath core port

Vlan  Group Address      Ver  Type  Port list
111   225.1.110.112      v2   D     Po110
```

2. From the mroute entry we see the hash algorithm has selected BL-acorn6 as the path to the RP. You will observe the CE VLAN 111 added to the OIF, along with the **igmp** flag denoting that this OIF was created via an IGMP join. The IIF for the LHR (*,G) will always be the Layer-3 core VLAN as path to the RP will be through the fabric, towards the border leaf switch(es).

```
acorn3%PineForest:Tree1# show ip mroute 225.1.110.112
IP Multicast Routing Table for VRF "PineForest:Tree1

(*,225.1.110.112/32), uptime: 1w6d, igmp ip pim
  Incoming interface: Vlan1502, RPF nbr: 100.2.0.106
  Outgoing interface list: (count: 1)
    Vlan111, uptime: 1w4d, igmp
```

3. The mroute table will now update the fabric multicast table with the respective (*,G) entry for 225.1.110.112. In the fabric multicast table, we also store the chosen RPF neighbor, acorn6.

```
acorn3%PineForest:Tree1# show fabric multicast ipv4 mroute 225.1.110.112
VRF "PineForest:Tree1" Fabric mroute Database VNI: 50010

Fabric Mroute: (*, 225.1.110.112/32)
  Interested Fabric Nodes:
    This node    RPF Neighbor: 100.2.0.106
```

4. From the BGP output we see the (*,G) update from fabric multicast table. The next hop is 0.0.0.0, because acorn3 is itself the next hop to the receiver. BGP will now advertise the fabric mroute via SAFI messages to route reflector(s) located on the spine switch(es).

```
acorn3%PineForest:Tree1# show bgp ipv4 mvpn rd 60100:50010
BGP routing table information for VRF default, address family IPv4 MVPN
BGP table version is 625, local router ID is 100.2.0.103
Status: s-suppressed, x-deleted, S-stale, d-dampened, h-history, *-valid, >-best
Path type: i-internal, e-external, c-confed, l-local, a-aggregate, r-redist, I-injected
Origin codes: i - IGP, e - EGP, ? - incomplete, | - multipath, & - backup


Route Distinguisher: 60100:50010    (C-Multicast Join)
*>l[0.0.0.0][225.1.110.112][60100]/96
                     0.0.0.0                           100      32768 i
```

Route Reflector Verification

5. On the route reflector, we have no knowledge of the client VRF. To review that BGP has received the update from leaf switch acorn3, you can look at it based on the RD info (AS:VNI). From the output we can see that acorn3 has an interested receiver. The route reflector will advertise this BGP update to all leaf switches in the fabric.

```
Oak1# show bgp ipv4 mvpn rd 60100:50010
BGP routing table information for VRF default, address family IPv4 MVPN
BGP table version is 942, local router ID is 100.2.0.1
Status: s-suppressed, x-deleted, S-stale, d-dampened, h-history, *-valid, >-best
Path type: i-internal, e-external, c-confed, l-local, a-aggregate, r-redist, I-injected
Origin codes: i - IGP, e - EGP, ? - incomplete, | - multipath, & - backup

Route Distinguisher: 60100:50010    (C-Multicast Join)
* i[0.0.0.0][225.1.110.112][60100]/96
*>i                  100.2.0.103                     100          0 i
```

Border Leaf Switch RP Verification

6. Once BGP SAFI update reaches the border leaf switches, we confirm acorn3 is added as the next-hop, and then confirm the fabric multicast and mroute tables are updated respectively.
   Note: You will see two entries in the BGP MVPN table, 1 from each route reflector in the topology.

```
acorn6%PineForest:Tree1# show bgp ipv4 mvpn rd 60100:50010
BGP routing table information for VRF default, address family IPv4 MVPN
BGP table version is 440, local router ID is 100.2.0.106
Status: s-suppressed, x-deleted, S-stale, d-dampened, h-history, *-valid, >-best
Path type: i-internal, e-external, c-confed, l-local, a-aggregate, r-redist, I-injected
Origin codes: i - IGP, e - EGP, ? - incomplete, | - multipath, & - backup
   Network            Next Hop            Metric     LocPrf     Weight Path
Route Distinguisher: 60100:50010    (C-Multicast Join)
*>i[0.0.0.0][225.1.110.112][60100]/96
                      100.2.0.103                    100          0 i
* i                   100.2.0.103                    100          0 i

acorn6%PineForest:Tree1# show fabric multicast ipv4 mroute 225.1.110.112
VRF "PineForest:Tree1" Fabric mroute Database VNI: 50010

Fabric Mroute: (*, 225.1.110.112/32)
  Interested Fabric Nodes:
    100.2.0.103 (real) (aggr)    RPF Neighbor: 100.2.0.106
```

7. The (*,G) mroute entry is built based on the fabric multicast update denoted by **fabric_mcast** flag. Because the receiver is in the fabric, the Layer-3 core VLAN will be the OIF, and the IIF will be the DCI sub-interface pointing to the RP.
   Important - Note that the Layer-3 core VLAN on the border leaf switch acorn106 is 1510 and differs from acorn3 which has VLAN 1502. The Layer-3 core VLAN is locally significant; the key is to make sure the VNI matches for both.

```
acorn6%PineForest: Tree1# show ip mroute detail
IP Multicast Routing Table for VRF "PineForest:Tree1"

(*, 225.1.110.112/32), uptime: 01:13:19, fabric_mcast ip pim
  Incoming interface: port-channel10.100, RPF nbr: 192.168.1.101
  Outgoing interface list: (count: 1) (Fabric OIF)
    Vlan1510, uptime: 01:13:19, fabric_mcast

acorn6%PineForest:Tree1# show ip pim route
PIM Routing Table for VRF "PineForest:Tree1" - 5 entries

(*, 225.1.110.112/32), RP 100.100.100.100*, expires 00:02:13, RP-bit
  Incoming interface: port-channel10.100, RPF nbr 192.168.1.101
  Oif-list: (0) 00000000, timeout-list: (0) 00000000
  Timeout-interval: 3, JP-holdtime round-up: 3

acorn3%PineForest:Tree1# show vlan id 1502 vn-segment

VLAN Segment-id
---- -----------
1502 50010

acorn6%PineForest:Tree1# show vlan id 1510 vn-segment

VLAN Segment-id
---- -----------
1510 50010
```

8. The same verification steps should be used to verify BGP (S,G) join propagation as well. Once the LHR receives multicast data traffic for 225.1.110.112, it will (1) create the (S,G) entry in the mroute table, (2) build a BGP (S,G)

join, and (3) repeat the same process as described above for (*,G) state. The primary difference will be that LHR acorn3 will select an RPF neighbor of the FHR leaf switch (if the source is inside the fabric), or again for one of the border leaf switches (if the source is outside of the fabric).

In situations where a receiver is located inside the fabric and is interested in a group sourced outside of the fabric, the IGMP join will be translated to a BGP Join at the LHR leaf switch, and that BGP Join will be translated into a PIM Join at the border leaf switch and forwarded to the RPF PIM neighbor outside of the fabric. If there are redundant border leaf switches and the source is outside the fabric, only the border leaf switch which is determined as the *fabric forwarder* for the respective group will forward the multicast traffic into the fabric to prevent duplicates.

You have now successfully verified BGP join propagation through the fabric from server leaf switch to the border leaf switches, and have the commands needed to be able to isolate which device is having a problem if troubleshooting is needed.

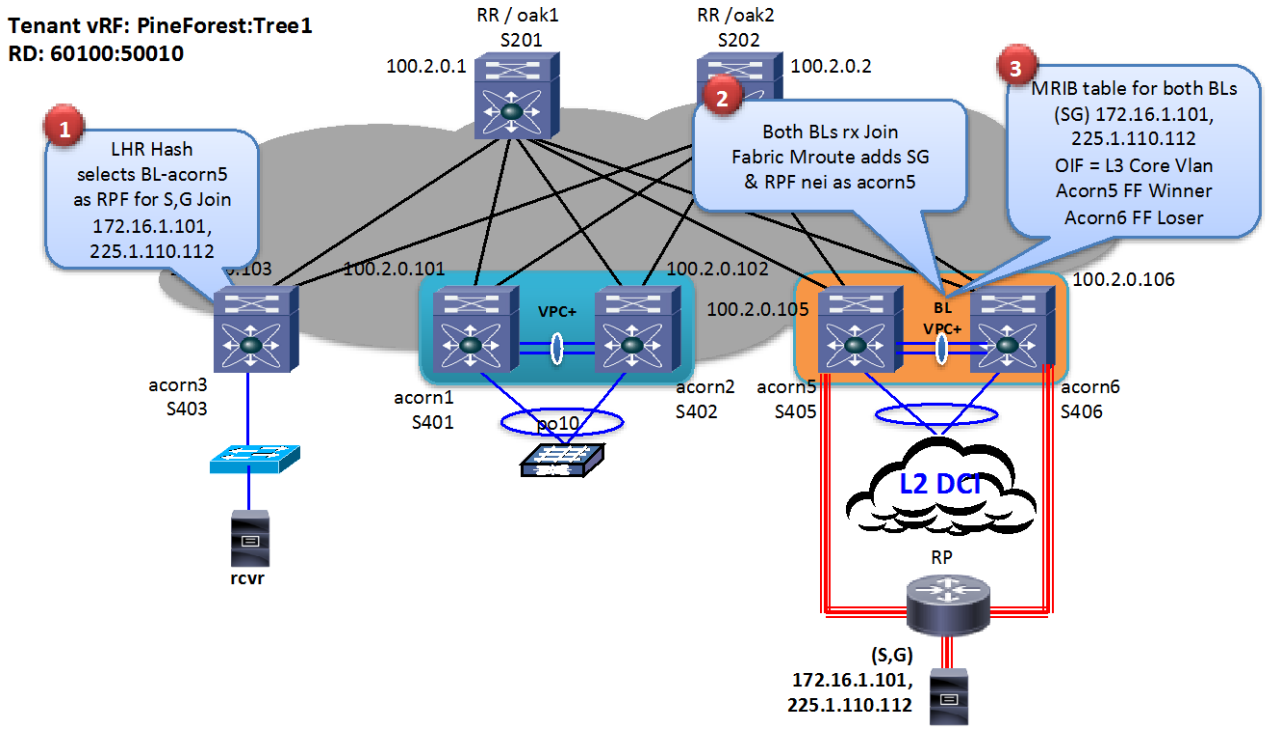
5.1.3 Fabric Forwarder Election for Redundant Border Leaf Switches

When there are redundant border leaf switches and a source outside of the fabric, the border leaf switch that will be responsible for forwarding the multicast traffic into the fabric is known as the *fabric forwarder* (do not confuse this with *vPC+ forwarder* which is discussed later). The *fabric forwarder* is determined on a per (S,G) basis. This means that both border leaf switches could be fabric forwarders for different (S,G)s. The fabric forwarder for a specific (S,G) is determined by a hash algorithm on the LHR leaf switch that is used to specify which border leaf switch the BGP (S,G) Join should be destined to. Both border leaf switches will actually receive this (S,G) Join, however only the border leaf switch in which the join is destined to (based on the hash algorithm on the LHR) will be elected the Fabric Forwarder and forward the (S,G) traffic into the fabric.

Note: The hash algorithm procedures are similar to the upstream-multicast hop (UMH) selection (per RFC 6513).

In this example we will verify whether acorn6 or acorn5 is elected as fabric forwarder for (S,G) (172.16.1.101, 225.1.110.112), which is sourced behind the Layer-3 edge router outside of the fabric.

High-level Overview for Fabric Forwarder Election



Assuming that LHR acorn3 receives the multicast traffic, shortest path tree (SPT) cutover is triggered and builds the subsequent BGP (S,G) join.

1. LHR acorn3 will utilize a hash to determine which border leaf switch should be used as the RPF neighbor for the (S,G) (172.16.1.101, 225.1.110.112). In this case it has chosen acorn5.
2. Both border leaf switches will receive the BGP Join, and populate fabric multicast table with the (S,G) entry, and the selected RPF neighbor.
3. Mroute table (MRIB) will be updated on both border leaf switches with the (S,G), but only acorn5 which was chosen as the (S,G) RPF neighbor, will become the fabric forwarder for that (S,G). The other border leaf switch, acorn6, will be marked *fabric forwarder loser*. As a result, in topologies where both border leaf switches receive the traffic, only the fabric forwarder will route the traffic into the fabric.

Detailed Troubleshooting and Verification Steps for Fabric Forwarder Election

1. In the previous section we observed that for (*, 225.1.110.112), the LHR acorn3 hash selected RPF neighbor acorn6. Now, verify which RPF neighbor the LHR selects for the (S,G) join of (172.16.1.101, 225.1.110.112). We notice in the example below that it will choose the other border leaf switch acorn5. The hash algorithm on the LHR can select the same border leaf switch or a different one for the (*,G), and (S,G) joins since the hash is done on different source addresses.

```
acorn3%PineForest:Tree1(config)# show fabric multicast ipv4 mroute 225.1.110.112
VRF "PineForest:Tree1" Fabric mroute Database VNI: 50010

Fabric Mroute: (*, 225.1.110.112/32)
  Interested Fabric Nodes:
```

```
    This node    RPF Neighbor: 100.2.0.106

Fabric Mroute: (172.16.1.101/32, 225.1.110.112/32)
  Interested Fabric Nodes:
    This node    RPF Neighbor: 100.2.0.105
```

2. Both border leaf switches should receive this BGP (S,G) Join for (172.16.1.101, 225.1.110.112) via the update from both BGP route reflectors. They will then update their BGP tables and add the selected RPF neighbor information into the fabric multicast table.

```
acorn5%PineForest:Tree1(config)# show bgp ipv4 mvpn rd 60100:50010
BGP routing table information for VRF default, address family IPv4 MVPN
BGP table version is 537, local router ID is 100.2.0.105

   Network            Next Hop            Metric     LocPrf     Weight Path
Route Distinguisher: 60100:50010    (C-Multicast Join)
* i[0.0.0.0][225.1.110.112][60100]/96
                     100.2.0.103                      100          0 i
*>i                  100.2.0.103                      100          0 i

Route Distinguisher: 60100:50010    (C-Multicast Join)
*>i[172.16.1.101][225.1.110.112][60100]/96
                     100.2.0.103                      100          0 i
* i                  100.2.0.103                      100          0 i


acorn5%PineForest:Tree1(config)# show fabric multicast ipv4 mroute 225.1.110.112
VRF "PineForest:Tree1" Fabric mroute Database VNI: 50010

Fabric Mroute: (*, 225.1.110.112/32)
  Interested Fabric Nodes:
    100.2.0.103 (real) (aggr)    RPF Neighbor: 100.2.0.106

Fabric Mroute: (172.16.1.101/32, 225.1.110.112/32)
  Interested Fabric Nodes:
    100.2.0.103 (real)    RPF Neighbor: 100.2.0.105


acorn6%PineForest:Tree1(config)# show bgp ipv4 mvpn rd 60100:50010
BGP routing table information for VRF default, address family IPv4 MVPN
BGP table version is 85, local router ID is 100.2.0.106

   Network            Next Hop            Metric     LocPrf     Weight Path
Route Distinguisher: 60100:50010    (C-Multicast Join)
* i[0.0.0.0][225.1.110.112][60100]/96
                     100.2.0.103                      100          0 i
*>i                  100.2.0.103                      100          0 i

Route Distinguisher: 60100:50010    (C-Multicast Join)
*>i[172.16.1.101][225.1.110.112][60100]/96
                     100.2.0.103                      100          0 i
* i                  100.2.0.103                      100          0 i


acorn6%PineForest:Tree1(config)# show fabric multicast ipv4 mroute 225.1.110.112
VRF "PineForest:Tree1" Fabric mroute Database VNI: 50010
```

```
Fabric Mroute: (*, 225.1.110.112/32)
  Interested Fabric Nodes:
    100.2.0.103 (real) (aggr)    RPF Neighbor: 100.2.0.106

Fabric Mroute: (172.16.1.101/32, 225.1.110.112/32)
  Interested Fabric Nodes:
    100.2.0.103 (real)    RPF Neighbor: 100.2.0.105
```

3. Using the mroute table, we can now verify acorn5 was elected the fabric forwarder winner and acorn6 was elected fabric forwarder loser (technically we only mark the loser and the winner is implicit from the loser notation on the peer). However, pay close attention to these mroute table entries for both tables.

   Because acorn6 is chosen as the (*,G) RPF neighbor, it will then create the (*,G) in it's respective mroute table with the Layer-3 core VLAN as the OIF. Acorn6 also received the BGP (S,G) Join for (172.16.1.101, 225.1.110.112), and it will also create the (S,G) entry in the mroute table, not because it was chosen as the RPF neighbor (as acorn5 was), but because it has an existing (*,G) entry for the same group. So acorn6 will create the (*,G) and (S,G). Acorn6 will also even add the Layer-3 core VLAN to the OIF table as well, however it does not add the OIF because it is the forwarder, but instead because it is copying the OIF from the (*,G). Notice in the example below that the *mrib* flag beside its OIF entry. This OIF was added because of the standard multicast rule that all (*,G) OIFs are copied down to (S,G) OIFs. So even though border leaf switch acorn6 has the Layer-3 core VLAN in the OIF list, it will be marked as the *fabric forwarding loser* and not allowed to forward packets received for this (S,G) into the fabric (because it was not selected as the (S,G) RPF neighbor).

   acorn5 was not selected as the (*,G) RPF neighbor, so it will not create a (*,G) entry in the mroute table. However, it was selected as the RPF neighbor for the (S,G), therefore creating the (S,G) entry in the mroute table with the OIF of the Layer-3 core VLAN, and become the fabric forwarding winner**.** In the outputs below, take note of the flags by each respective OIF entry, mrib vs fabric_mcast as mentioned above.

```
acorn6%PineForest:Tree1(config)# show ip mroute 225.1.110.112
IP Multicast Routing Table for VRF "PineForest:Tree1"

(*, 225.1.110.112/32), uptime: 01:18:18, fabric_mcast ip pim
  Incoming interface: port-channel10.100, RPF nbr: 172.16.1.101
  Outgoing interface list: (count: 1) (Fabric OIF)
    Vlan1510, uptime: 01:18:18, fabric_mcast

(172.16.1.101/32, 225.1.110.112/32), uptime: 00:42:04, fabric_mcast ip mrib pim
  Incoming interface: port-channel10.100, RPF nbr: 172.16.1.101
  Outgoing interface list: (count: 1) (Fabric Forwarding Loser)
    Vlan1510, uptime: 00:42:02, mrib


acorn5%PineForest:Tree1(config)# show ip mroute 225.1.110.112
IP Multicast Routing Table for VRF "PineForest:Tree1"

(172.16.1.101/32, 225.1.110.112/32), uptime: 00:42:04, fabric_mcast ip pim
  Incoming interface: port-channel10.100, RPF nbr: 172.16.1.101
  Outgoing interface list: (count: 1) (Fabric OIF)
    Vlan1510, uptime: 00:42:02, fabric_mcast
```

4. Both border leaf switches having the Layer-3 core VLAN added to the OIF is a result of LHR acorn3 hash selecting two different RPF neighbors for the (*,G) and (S,G). As a result of both border leaf switches having the OIF, we need to elect the fabric forwarder. If acorn3 had chosen the same RPF neighbor for the (*,G) and (S,G), then the end result would be that only that border leaf switch chosen for both would have an OIF. Thus no border leaf switch would be marked as the fabric forwarder loser since only one border leaf switch has the OIF, and there is no possibility of both border leaf switches forwarding into the fabric.

   To summarize the two possible scenarios:
   (1) Both border leaf switches will have OIF, and one is denoted as fabric forwarder loser when different border leaf switches are chosen as (*,G) and (S,G) RPF neighbor by the LHR hash.
   (2) Only one border leaf switch has the OIF. This is when the same border leaf switch is chosen as (*,G) and (S,G) RPF neighbor by the LHR hash.

You have now successfully verified fabric forwarder election with redundant border leaf switches and a source outside of the fabric. The commands above can be used to isolate a problem if troubleshooting is required.

5.1.4 vPC+

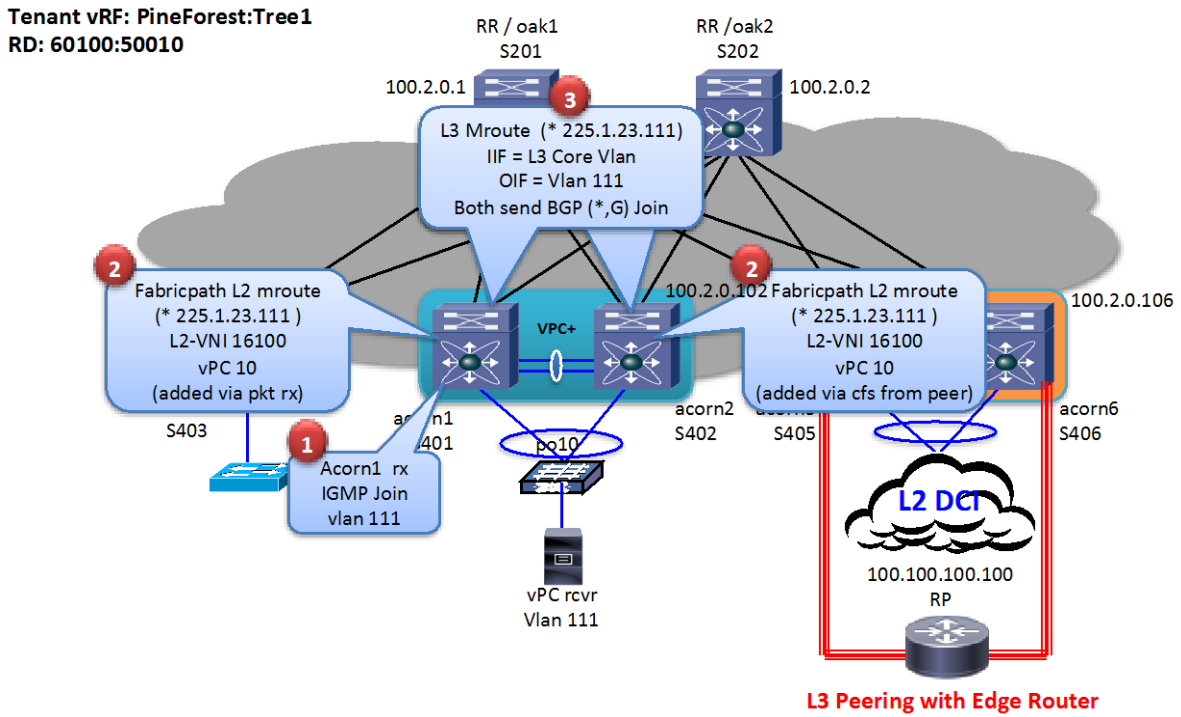
In vPC+ implementations within the fabric, it is always important to understand which peer is responsible for FHR source registration, sending BGP joins, and forwarding the traffic when we have sources and/or receivers behind a vPC or orphan interface. With vPC+ inside the fabric, we do not have a concept of *Designated Router* source registration. If a source is behind a vPC interface, then whichever peer receives the traffic (due to the downstream devices channel hash) will then be the device that does the FHR source registration. In Cisco Nexus 5000 and 6000 Series switches vPC+ scenarios, the multicast data traffic, or IGMP control packet is not forwarded across the vPC+ peer-link (this is different than Cisco Nexus 7000 Series switch implementations, where the data traffic and IGMP packets are forwarded across the peer link). Therefore, in cases where a receiver is behind a vPC, then whichever switch receives the join will synchronize this information to its peer using Cisco Fabric Services (CFS) messages. Both vPC+ peers will have this join information and forward the appropriate BGP Join messages upstream. Though both devices will send a BGP Join to pull the traffic to the vPC+, only one of them will forward the packets to a receiver connected to a vPC so that we prevent duplicates. Details on how this is determined are explained in the *Multicast Data-Plane Forwarding* section. If a join is received from an orphan port, then there will be no synchronizing of information between peers.

Let's look at both scenarios: (1) A receiver behind vPC+ and (2) A source behind vPC+.

Scenario 1: Receiver behind vPC+

In this example, we will focus on how an IGMP Join for 225.1.23.111 on VLAN 111 / VNI-16100 is handled once received on a vPC interface.
The vPC+ pair is made up of acorn1 and acorn2, (with an emulated Switch-ID of 501). We will not go into forwarding the join through the fabric, as that was covered in detail in the *BGP Join Creation and Propagation* section.

High-level Overview for Receiver behind vPC+



1. The downstream device will determine which link/vPC+ peer receives the IGMP Join. In this case, acorn1 is the recipient.
2. This switch will program the join received on the Layer-2 VNI 16100 into its fabricpath Layer-2 mroute table, and recognizing that the IGMP join was received on a vPC interface. This triggers a CFS message to its vPC peer informing it to program the same information so they are in sync. (As stated previously, Cisco Nexus 5000 and 6000 Series switches implementations will not forward the IGMP packet or multicast data packets across the vPC+ Peer-Link)
3. Both vPC+ peers will also create (*,G) state with the OIF of the VLAN the join was received on, and initiate (*,G) BGP Joins to the RP. Note that even though both have the OIF, only one will forward the packet. This is covered in detail in the *Multicast Data-Plane Forwarding* section.

Detailed Troubleshooting and Verification Steps for Receiver behind vPC+

1. The downstream device will determine which interface the join hashes to. The IGMP join will be added to the IGMP snooping table for VLAN 111. No matter which device receives the actual packet, the information will be synced to its peer via CFS messages across the peer-link, even though the original IGMP join packet will terminate on the receiving vPC+ leaf.

```
acorn1%PineForest:Tree1# show ip igmp snooping groups 225.1.23.111 vlan 111
Type: S - Static, D - Dynamic, R - Router port, F - Fabricpath core port

Vlan  Group Address      Ver  Type  Port list
111   225.1.23.111       v2   D     Po10

acorn2%PineForest:Tree1# show ip igmp snooping groups 225.1.23.111 vlan 111
```

```
Type: S - Static, D - Dynamic, R - Router port, F - Fabricpath core port

Vlan  Group Address      Ver  Type  Port list
111   225.1.23.111       v2   D     Po10
```

Note: You cannot track CFS updates on a per-group basis. However, if vPC peers are out of sync, you can verify that CFS messages are properly being sent and received via the following command.

```
acorn1%PineForest:Tree1# show ip igmp snooping statistics vlan 111

VLAN 111 IGMP snooping statistics, last reset: never (only non-zero values displayed)
  Packets received: 418
  IGMPv2 reports received: 209
  IGMPv2 queries received: 209
  PIM Hellos received: 1737
  Packets sent to routers: 209
  VIM IGMP leave sent on failover: 0
  vPC Peer Link CFS packet statistics:
      IGMP packets (sent/recv/fail): 209/0/0

Acorn2%PineForest:Tree1# show ip igmp snooping statistics vlan 111

VLAN 111 IGMP snooping statistics, last reset: never (only non-zero values displayed)
  Packets received: 418
  IGMPv2 reports received: 209
  IGMPv2 queries received: 209
  PIM Hellos received: 1737
  VIM IGMP leave sent on failover: 0
  vPC Peer Link CFS packet statistics:
      IGMP packets (sent/recv/fail): 0/209/0
```

2. Once the IGMP join is received (either directly or via CFS from vPC peer), the Layer-2 VNI (16100) + group information (225.1.23.111) will be added to the fabricpath mroute table. Verify that the CE VLAN to Layer-2 VNI mapping is correct, and that the proper interface is added to the Layer-2 OIF list.

```
acorn1%PineForest:Tree1# show vlan id 111 vn-segment
VLAN Segment-id
---- -----------
111  16100


acorn2%PineForest:Tree1# show vlan id 111 vn-segment
VLAN Segment-id
---- -----------
111  16100

acorn1%PineForest:Tree1# show fabricpath mroute vni 16100

(vlan/16100, 0.0.0.0, 225.1.23.111), uptime: 00:54:48, igmp
 Outgoing interface list: (count: 1)
  Interface port-channel10, uptime: 00:54:48, igmp


acorn2%PineForest:Tree1# show fabricpath mroute vni 16100
```

```
(vlan/16100, 0.0.0.0, 225.1.23.111), uptime: 00:54:49, igmp
 Outgoing interface list: (count: 1)
  Interface port-channel10, uptime: 00:54:49, igmp
```

3. Aside from the Layer-2 tables being populated above, below you will see the Layer-3 mroute table should also be populated with a (*,G) entry for 225.1.23.111. The incoming interface for the (*,G) should always be the Layer-3 core VLAN, and the outgoing interface will be the CE VLAN for which the join was received on, VLAN 111. Both vPC+ peers will create the same mroute table state.

```
acorn1%PineForest:Tree1# show ip mroute 225.1.23.111
IP Multicast Routing Table for VRF "PineForest:Tree1"

(*, 225.1.23.111/32), uptime: 05:38:06, igmp ip pim
  Incoming interface: Vlan1502, RPF nbr: 100.2.0.105
  Outgoing interface list: (count: 1)
    Vlan111, uptime: 05:38:06, igmp


acorn2%PineForest:Tree1# show ip mroute 225.1.23.111
IP Multicast Routing Table for VRF "PineForest:Tree1"

(*, 225.1.23.111/32), uptime: 05:38:06, igmp ip pim
  Incoming interface: Vlan1502, RPF nbr: 100.2.0.105
  Outgoing interface list: (count: 1)
    Vlan111, uptime: 05:38:06, igmp
```

4. Both vPC+ peers will also update their fabric multicast and BGP tables with information that they both have an interested receiver for 225.1.23.111. With both peers' BGP tables updated, they will then send a BGP Join into the fabric. In the output below, the fabric mroute "This node" equates to BGP table 0.0.0.0 as the next hop, both basically saying this device is originating this (*,G) Join.

```
acorn1%PineForest:Tree1# show fabric multicast ipv4 mroute 225.1.23.111
VRF "PineForest:Tree1" Fabric mroute Database VNI: 50010

Fabric Mroute: (*, 225.1.23.111/32)
  Interested Fabric Nodes:
    This node    RPF Neighbor: 100.2.0.105

acorn2%PineForest:Tree1# show fabric multicast ipv4 mroute 225.1.23.111
VRF "PineForest:Tree1" Fabric mroute Database VNI: 50010

Fabric Mroute: (*, 225.1.23.111/32)
  Interested Fabric Nodes:
    This node    RPF Neighbor: 100.2.0.105

acorn1%PineForest:Tree1# show bgp ipv4 mvpn rd 60100:50010
BGP routing table information for VRF default, address family IPv4 MVPN
BGP table version is 584, local router ID is 100.2.0.101

   Network            Next Hop            Metric     LocPrf     Weight Path
Route Distinguisher: 60100:50010    (C-Multicast Join)

*>l[0.0.0.0][225.1.23.111][60100]/96
                      0.0.0.0                          100      32768 i
```

```
acorn2%PineForest:Tree1# show bgp ipv4 mvpn rd 60100:50010
BGP routing table information for VRF default, address family IPv4 MVPN
BGP table version is 584, local router ID is 100.2.0.102

   Network            Next Hop            Metric     LocPrf     Weight Path
Route Distinguisher: 60100:50010    (C-Multicast Join)

*>l[0.0.0.0][225.1.23.111][60100]/96
                      0.0.0.0                           100      32768 i
```

5. Since both vPC+ peers have originated a (*,G) BGP Join, they will both create the (S,G) 225.1.23.111 + OIF VLAN111, but only one of the two vPC+ peers will forward. How the device selected for forwarding is determined is covered in *Multicast Data-Plane Forwarding,* a subsequent section. In the end, for a receiver behind a vPC interface, the control plane state on both peers should look identical as shown below.

```
acorn1%PineForest:Tree1# show ip mroute
IP Multicast Routing Table for VRF "PineForest:Tree1"

(*, 225.1.23.111/32), uptime: 06:05:55, igmp ip pim
  Incoming interface: Vlan1502, RPF nbr: 100.2.0.105
  Outgoing interface list: (count: 1)
    Vlan111, uptime: 06:05:55, igmp

(10.1.1.109/32, 225.1.23.111/32), uptime: 06:00:08, ip mrib pim
  Incoming interface: Vlan1502, RPF nbr: 100.2.0.103
  Outgoing interface list: (count: 1)
    Vlan111, uptime: 06:00:08, mrib
```

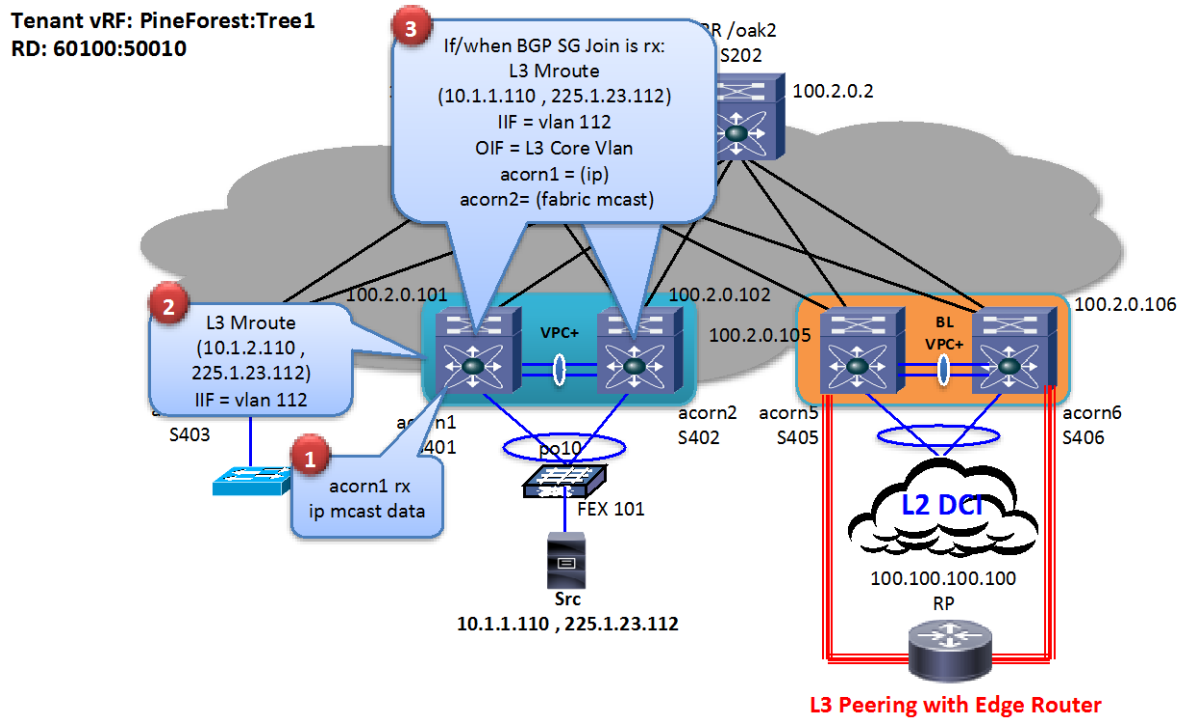
```
acorn2%PineForest:Tree1# show ip mroute
IP Multicast Routing Table for VRF "PineForest:Tree1"

(*, 225.1.23.111/32), uptime: 06:05:55, igmp ip pim
  Incoming interface: Vlan1502, RPF nbr: 100.2.0.105
  Outgoing interface list: (count: 1)
    Vlan111, uptime: 06:05:55, igmp

(10.1.1.109/32, 225.1.23.111/32), uptime: 06:00:08, ip mrib pim
  Incoming interface: Vlan1502, RPF nbr: 100.2.0.103
  Outgoing interface list: (count: 1)
    Vlan111, uptime: 06:00:08, mrib
```

You have now successfully verified IGMP join handling with a receiver behind vPC+ interface. The commands above can be used to isolate a problem if troubleshooting is required.

Scenario 2: Source behind vPC+

In this scenario, we will focus on a source behind a vPC interface, (10.1.2.110, 225.1.23.112). The location of the receiver is irrelevant when looking at how the vPC+ pair will handle the ingress multicast data traffic and create state.

High-level Overview for Source behind vPC+



1. Downstream device will determine which link/vPC+ peer receives the multicast data traffic. In this case, acorn1 receives the multicast traffic.
2. This vPC+ peer will create (S,G) state based on the data packet, with IIF of the source, VLAN 112 (no OIF is populated until an interested receiver is present).
3. DFA utilizes standard PIM register process (multicast packets will be encapsulated in unicast and sent to the RP). Whether the RP is the border leaf switch, or beyond the border leaf switch outside the fabric, the process will be the same. (PIM register process is outside the scope of this whitepaper)
4. Once a remote receiver (receiver not on acorn1) eventually sends an (S,G) BGP Join, both vPC+ peers will receive it. No matter which peer the BGP Join is actually destined to, both vPC+ peers will accept the join based on the rule that they will accept a BGP Join destined to "me, or my vPC+ peer". Since acorn1 already created the (S,G) state from the IP packet, it will simply add the OIF (the Layer-3 core VLAN 1502). However, its vPC+ peer (acorn2), did not receive the packet, and thus will not never create (S,G) state. Acorn2 will create state based on reception of the BGP (S,G) Join, and it will also add the core VLAN (1502) as the OIF along with the source VLAN 112 as the IIF. The end result is that both vPC+ peers will have (S,G) state + OIFs (one from ip packet, the other from BGP join), but only the device that is receiving the traffic can forward it through the fabric (both peers create state + OIF because we cannot dictate which peer will be receiving traffic at any point in time).

Key Note: In NXOS, when (S,G) state is created by a data packet, it is denoted by *ip* in the (S,G) entry, and when created via BGP Join, it is denoted by *fabric mcast*.

Detailed Troubleshooting and Verification Steps when a Source is behind vPC+

1. acorn1 has created (S,G) state due to the native multicast traffic hashing to this switch from the downstream vPC switch interface. The *ip* notation in the mroute flags indicates this was created via an IP packet. The IIF will be the

CE VLAN 112 that the multicast stream was received on. There will be no OIF until an interested receiver comes online.

```
acorn1%PineForest:Tree1# show ip mroute 225.1.23.112
IP Multicast Routing Table for VRF "PineForest:Tree1"

(10.1.2.110/32, 225.1.23.112/32), uptime: 00:28:04, ip pim
Incoming interface: Vlan112, RPF nbr: 10.1.2.110
Outgoing interface list: (count: 0)
```

2. Once the border leaf switch is aware of an interested receiver [via a (*,G) join], the border leaf switch will eventually select an RPF neighbor for the (S,G) join towards the source. Acorn5 in this scenario has selected acorn1. Even though the BGP Join will be destined to Acorn1, it will actually be forwarded to all leaf switches in the fabric.

```
acorn5%PineForest:Tree1# show fabric multicast ipv4 mroute 225.1.23.112
VRF "PineForest:Tree1" Fabric mroute Database VNI: 50010

Fabric Mroute: (10.1.2.110/32, 225.1.23.112/32)
  Interested Fabric Nodes:
    This node    RPF Neighbor: 100.2.0.101
```

3. On the reception of the BGP Join from acorn5, acorn2 will create the (S,G) and both peers will program the OIF to the Layer-3 core VLAN 1502. No matter which vPC peer the BGP Join is actually destined to, both vPC+ peers will accept the Join for *itself* and *its peer* with the rule *"accept Join destined for myself or my vPC peer".* You can see this by the fabric_mcast flag set on the OIF of acorn1 and mroute on acorn2.

```
acorn1%PineForest:Tree1(config)# show ip mroute 225.1.23.112
IP Multicast Routing Table for VRF "PineForest:Tree1"

(10.1.2.110/32, 225.1.23.112/32), uptime: 00:28:04, ip pim fabric_mcast
  Incoming interface: Vlan112, RPF nbr: 10.1.2.110
  Outgoing interface list: (count: 1) (Fabric OIF)
    Vlan1502, uptime: 00:21:27, fabric_mcast

acorn2%PineForest:Tree1(config)# show ip mroute 225.1.23.112
IP Multicast Routing Table for VRF "PineForest:Tree1"

(10.1.2.110/32, 225.1.23.112/32), uptime: 00:21:27, fabric_mcast ip
  Incoming interface: Vlan112, RPF nbr: 10.1.2.110
  Outgoing interface list: (count: 1) (Fabric OIF) (Fabric PeerFwder)
    Vlan1502, uptime: 00:21:27, fabric_mcast
```

Key Note: In Cisco Nexus 5000 and 6000 Series switches implementations, **Fabric PeerFwder** can be ignored. This is a platform dependent notation specific to the Cisco Nexus 7700/7000 Series switches.

4. Both switches now have (S,G) state with an OIF entry of the Layer-3 core VLAN in order to send the multicast stream to the receiver. Only the peer that is receiving the multicast stream from the downstream vPC will forward into the fabric. To verify which device is receiving the packets, you can review the mroute counters below. However note that these are software counters which simply show which device created the (S,G) state via data packet reception, and not hardware data forwarding counters.

```
acorn1%PineForest:Tree1# show system internal mfwd ip mroute
MCASTFWD Multicast Routing Table for VRF "PineForest:Tree1"
(10.1.2.110/32, 225.1.23.112/32), data-created
  Software switched packets: 19, bytes: 1596

acorn2%PineForest:Tree1# show system internal mfwd ip mroute
MCASTFWD Multicast Routing Table for VRF "PineForest:Tree1"
(10.1.2.110/32, 225.1.23.112/32), data-created
  Software switched packets: 0, bytes: 0
```

You have now successfully verified control plane functioning when a source or receiver is located behind a vPC interface, and are able to isolate where a problem may be if troubleshooting is required.

5.2 Multicast Forwarding Data-Plane

Multicast data packet forwarding, just like unicast packet forwarding, is achieved using VN-segment aware FabricPath encapsulation.

Cisco DFA unified fabrics supports VLAN scaling beyond the 4000 VLANs traditionally offered by the 12 bit IEEE 802.1Q header in an Ethernet frame. By extending the traditional header value within the fabric, Cisco DFA has a 24-bit name space to uniquely address routing instances (VRF) and fabric global bridge domains (logical Layer-2 domains such as VLANs). This extended name space is achieved by VN-Segment IDs (VNIs).

For more information, refer to Cisco Dynamic Fabric Automation Technology: Overview and Deployment Considerations.

This extended name space removes the requirement for global VLANs and allows them to be locally significant to a leaf switch or a group of leaf switches. The stitching of local VLANs with the global VNI allows the extension of Layer 2 domains across the fabric in conjunction with enhanced forwarding and routing within and between subnets.

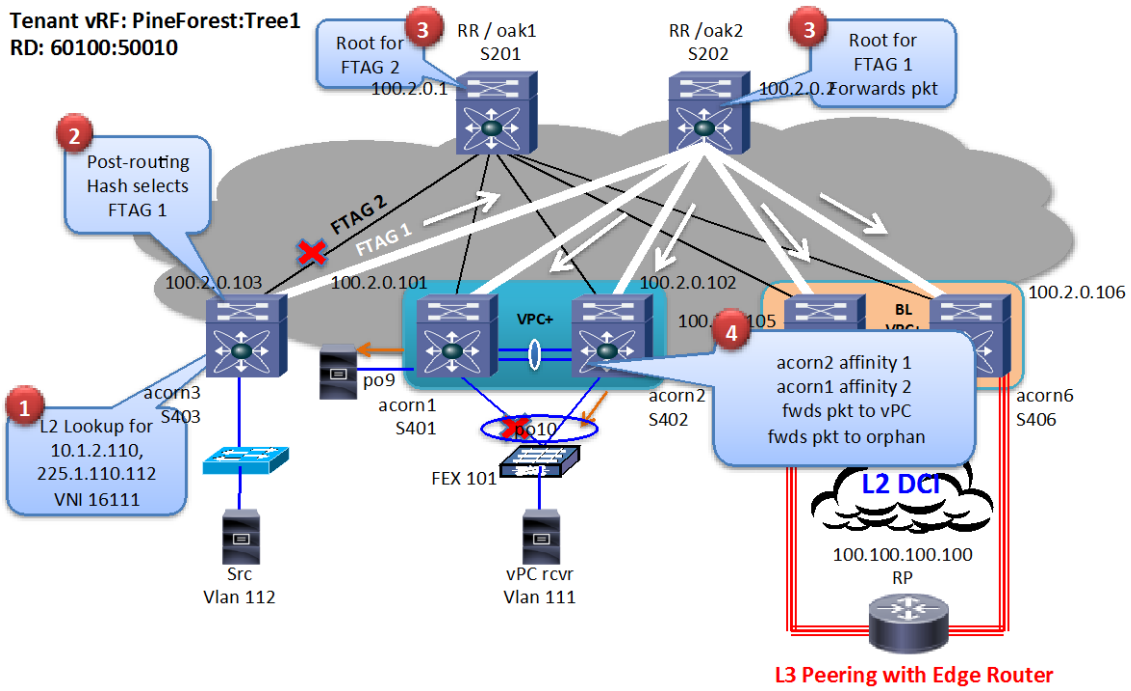
In DFA environments, rendezvous point (RP) reachability is required for multicast traffic to be forwarded through the fabric. All multicast traffic that traverses one leaf switch to another leaf switch is routed, even if that traffic is sourced and received in the same Layer-2 VNI/ broadcast domain. The multicast traffic is routed onto the core dynamic VLAN for the respective VRF. This means that on any particular leaf switch, the mroute table's IIF (for server leaf switches) or OIF (for source leaf switches) will be the core dynamic VLAN.

Every tenant VRF is tied to a single unique Layer-3 VNI (though typically, multiple Layer-2 VNIs are associated with each tenant VRF). Core dynamic VLANs, which are tied to a Layer3-VNI, have no server facing ports. They are used to route traffic across the fabric from a server leaf switch to another server leaf or border leaf. DFA fabrics with a FabricPath underlay currently do not support pruning (or snooping) of multicast packets at the spine switch level, however snooping is still supported at the server leaf switch towards CE ports.

We will pick up where control-plane left off. Initial mroute state is created and we will be focusing on the path of the packet from the source to the receiver. In the following example, we will follow the data-plane forwarding of a multicast source attached to acorn3 (FHR), and for multicast group 225.1.110.112 destined to a vPC+ receiver on ESWID 23 (acorn1 and acorn2) in VLAN 111.We will also have an orphaned port receiver off of acorn1.

Note: The source in this example is attached to a standalone leaf switch inside the fabric, but the same operations would hold true for a source outside of the fabric, and the border leaf switch would perform the operations we will outline for acorn2 (keep in mind if you have redundant border leaf switches, the fabric forwarder election covered previously would also come into play).

High-level Overview for Data-Plane Forwarding



1. The source sends a multicast packet and the packet ingress on the FHR acorn3 will trigger a Layer-2 forwarding lookup for 225.1.110.112 on the Layer-2 VNI (16111) for local receivers.
2. Once the Layer-3 core VLAN is programmed as the OIF (from an interested receiver), acorn3 will use a hash to determine which FabricPath FTAG it will use to forward the multicast packet to the respective FTAG root. In this scenario, FTAG 1 has been selected on acorn3.
3. In our case, Oak1 is the root for FTAG 2 and Oak2 is the root for FTAG 1 so the packet will be forwarded only to Oak2, which will then forward the traffic to all leaf switches in the fabric.
4. Both VPC+ peers will receive the multicast stream and create (S,G) state, but only the vPC+ peer with affinity for the FTAG chosen by the FHR leaf switch will forward the stream to the receiver.

Note: In situations where there is a source behind a vPC interface *and* the receiver was an orphan port; if the traffic were to hash to acorn2 and the orphan port were on the peer acorn1, then we would not forward this traffic over the peer-link. Instead this traffic would be forwarded to the spine switch and back down to the vPC peer the same way as described in the scenario above where source is on standalone acorn3. However, the two major differences from the above functionality would be as follows:

a) Once the packet is received by acorn1, it will still pass RPF check even though it was received from the fabric with a source address of a directly connected source. This is because the receiving peer will detect that the source is a vPC VLAN that it owns. This is referred to as a *Dual-RPF check*, where a source from a directly connected vPC host address is accepted for RPF even though the traffic was ingress from the Layer-3 core VLAN.
b) The receiving peer will forward the packet to any orphan receiver regardless of FTAG affinity, as FTAG affinity is only relevant when egress a vPC interface.

Detailed Troubleshooting and Verification Steps for Data-Plane Forwarding

FHR Verification

1. Once acorn3 receives the multicast data packet ingress on VLAN 112, it will perform a Layer-2 FabricPath mroute lookup based on the *group + VN-segment* combination. In DFA, the user VLAN is significant only within a single mobility domain. The same VLAN in a different mobility domain could be a separate broadcast domain. Hence, the importance to perform this lookup on the Layer-2 VNI, which will be globally significant, and defines a broadcast domain in the fabric.

```
acorn3%PineForest:Tree1# show vlan id 112 vn-segment

VLAN Segment-id
---- -----------
112  16111


acorn3%PineForest:Tree1# show fabricpath mroute vni 16111

(vlan/16111, *, *), Router ports (OMF), uptime: 4d21h, igmp
 Outgoing interface list: (count: 1)
  Interface Vlan112, [SVI] uptime: 4d21h, igmp
```

2. Acorn3 will now also create the Layer-3 mroute (S,G) entry. The incoming interface will be CE VLAN 112 that received the traffic, whereas the OIF will be populated with VLAN 1502, the Layer-3 core VLAN for the respective VRF. This Layer-3 core VLAN should map to the VNI for the respective VRF as all multicast traffic in this VRF will be routed through this Layer-3 core VLAN as shown below.

```
acorn3%PineForest:Tree1# show ip mroute
IP Multicast Routing Table for VRF "PineForest:Tree1"

(10.1.2.110/32, 225.1.110.112/32), uptime: 01:48:32, ip pim fabric_mcast
  Incoming interface: Vlan112, RPF nbr: 10.1.2.110
  Outgoing interface list: (count: 1) (Fabric OIF)
    Vlan1502, uptime: 01:48:32, fabric_mcast

acorn3%PineForest:Tree1# show vlan id 1502 vn-segment
VLAN Segment-id
---- -----------
1502 50010


acorn3%PineForest:Tree1# show run vrf PineForest:Tree1
vrf context PineForest:Tree1
```

```
vni 50010
```

3. Once the packet is routed into VLAN 1502, forwarding through the fabric will be done via FabricPath FTAG. To determine which FTAG acorn3 will use to forward the traffic onto the fabric, use the following command:

Key Note: You will use the post-routed VLAN (Layer-3 core VLAN 1502) but "src-mac" will be the MAC address of the packet before rewrite.

```
acorn3%PineForest:Tree1# show fabricpath load-balance multicast ftag-selected vlan 1502 post-
routed dst-ip 225.1.110.112 src-ip 10.1.2.110 dst-mac 0100.5e01.6e70 src-mac 0000.0112.0112 vrf
PineForest:Tree1

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

Ftag for post-route: 1

 topo_id: 0
 num_ftags : 2

::Hash params used::
pkt_hash: 0xc5
--Polynomial 0x2f Degree 0x8
bd_hash: 0x9b
--Polynomial 0x31 Degree 0x8
Offset: 0
dst_ip: 225.1.110.112
src_ip: 10.1.2.110
dst_mac: 0100.5e01.6e70
src_mac: 0000.0112.0112
vlan: 1502 (int-vlan: 2510)

====================================
```

4. At this point, all forwarding decisions will be *FabricPath switched* though the fabric. FabricPath is used as the underlay and multi destination traffic will be based on the FTAG tree. Now that we understand which FTAG the multicast packet will be placed, we can determine the exact path through the fabric.

   a) Determine FabricPath root with the following command

```
acorn3%PineForest:Tree1# show fabricpath isis topology summary
FabricPath IS-IS Topology Summary
Fabricpath IS-IS domain: default
MT-0
  Configured interfaces:  Ethernet1/1  Ethernet1/2
 Max number of trees: 2  Number of trees supported: 2
    Tree id: 1, ftag: 1, root system: 8c60.4f54.3ec1, 114
    Tree id: 2, ftag: 2, root system: 8c60.4f54.3dc1, 113
```

   b) Determine which interfaces on acorn3 are set to forward traffic with FTAG 1, and which interface is the best path to the root for FTAG 1. The interface with the best metric (lowest metric is the preferred one) to the root.

```
acorn3%PineForest:Tree1# show fabricpath isis trees multidestination 1
Fabricpath IS-IS domain: default
Note: The metric mentioned for multidestination tree is from the root of that tree to that switch-id
*:directly connected neighbor or link
P:Physical switch-id, E:Emulated, A:Anycast

MT-0
Topology 0, Tree 1, Swid routing table
23, L1
 via Ethernet1/2, metric 40
109, L1
 via Ethernet1/2, metric 40
111, L1
 via Ethernet1/2, metric 40
112, L1
 via Ethernet1/2, metric 40
113, L1
 via Ethernet1/2, metric 80
114, L1
 via Ethernet1/2, metric 0
```

c) Now that we know which interface will forward the multicast data packets out to the spine switch or BGP route reflector, confirm that hardware programming has occurred on acorn3 for group 225.1.110.112, as the Nexus platforms only forward multicast packets in hardware.

```
acorn3%PineForest:Tree1# show system internal forwarding vrf pineForest:Tree1 multicast route group
225.1.110.112

Multicast routes for table PineForest:Tree1/base
Dev | Index     | Group              | Source             | MET Ptr | RPF Int/
    |           |                    |                    |         | B-RP ord
----+-----------+--------------------+--------------------+---------+----------
1    0x148b      225.1.110.112/32     10.1.2.110/32        0x2       0x7a
```

Note: The multicast stream will be forwarded to all leaf switches in the fabric. In this scenario, the receiver is off the vPC leaf switches. Both switches will receive the stream! We will focus on how to determine which peer will forward the traffic to the host in the following section.

Last Hop Router Verification (vPC Receiver)

5. Both leaf switches will have (*,G) and (S,G) state (details previously explained in vPC+ control plane section). The (*,G) was initially created by the IGMP join from the local receiver in VLAN 111, and the (S,G) is built based on the reception of the multicast stream. As the OIF for the FHR acorn3 was the Layer-3 core VLAN, the IIF for the LHR should also be the Layer-3 core VLAN, and also, map to the same VNI for the VRF, in this case VNI 50010.

```
acorn1%PineForest:Tree1# show ip mroute 225.1.110.112
IP Multicast Routing Table for VRF "PineForest:Tree1"
```

```
(*, 225.1.110.112/32), uptime: 1d07h, igmp ip pim
  Incoming interface: Vlan1502, RPF nbr: 100.2.0.105
  Outgoing interface list: (count: 1)
    Vlan111, uptime: 1d07h, igmp

(10.1.2.110/32, 225.1.110.112/32), uptime: 02:06:56, ip mrib pim
  Incoming interface: Vlan1502, RPF nbr: 100.2.0.103
  Outgoing interface list: (count: 1)
    Vlan111, uptime: 02:06:56, mrib

acorn2%PineForest:Tree1# show ip mroute 225.1.110.112
IP Multicast Routing Table for VRF "PineForest:Tree1"

(*, 225.1.110.112/32), uptime: 1d07h, igmp ip pim
  Incoming interface: Vlan1502, RPF nbr: 100.2.0.105
  Outgoing interface list: (count: 1)
    Vlan111, uptime: 1d07h, igmp

(10.1.2.110/32, 225.1.110.112/32), uptime: 02:06:35, ip mrib pim
  Incoming interface: Vlan1502, RPF nbr: 100.2.0.103
  Outgoing interface list: (count: 1)
    Vlan111, uptime: 02:06:35, mrib



acorn1%PineForest:Tree1# show vlan id 1502 vn-segment
VLAN Segment-id
---- -----------
1502 50010

acorn2%PineForest:Tree1# show vlan id 1502  vn-segment
VLAN Segment-id
---- -----------
1502 50010
```

6. Since we copy all OIFs from (*,G) to (S,G), both vPC+ peers will also have an OIF for the respective (S,G) entry 10.1.2.110, 225.1.110.112. Since both vPC+ peers have an mroute with the OIF of VLAN 111, we need to verify that the Layer-2 FabricPath mroute table will show the expected vPC receiver and mapped to the Layer-2 VNI of VLAN 111.

```
acorn1%PineForest:Tree1# show vlan id 111 vn-segment
VLAN Segment-id
---- -----------
111  16100


acorn2%PineForest:Tree1# show vlan id 111 vn-segment
VLAN Segment-id
---- -----------
111  16100


acorn1%PineForest:Tree1# show fabricpath mroute vni 16100

(vlan/16100, 0.0.0.0, 225.1.110.112), uptime: 1d07h, igmp
 Outgoing interface list: (count: 2)
  Interface port-channel9, uptime: 00:19:16, igmp
```

```
  Interface port-channel10, uptime: 1d07h, igmp


acorn2%PineForest:Tree1# show fabricpath mroute vni 16100

(vlan/16100, 0.0.0.0, 225.1.110.112), uptime: 1d08h, igmp
 Outgoing interface list: (count: 1)
  Interface port-channel10, uptime: 1d08h, igmp
```

7. We now need to determine which vPC Peer will be responsible for forwarding the traffic to the receiver. In VPC+, this will be based on FTAG affinity. Only 1 of the peer switches will be responsible for one of the FTAG topologies. We can see that only acorn2 has affinity for FTAG-1, which was the FTAG chosen by the FHR acorn3 as the FTAG to be used to forward multicast data packets through the fabric. To determine affinity do the following:

```
acorn1%PineForest:Tree1# show fabricpath topology ftag active
Topo-Description          Topo-ID    Graph-ID  Ftag
------------------------- ---------- --------- ---------
0                         0          2         2

acorn2%PineForest:Tree1# show fabricpath topology ftag active
Topo-Description          Topo-ID    Graph-ID  Ftag
------------------------- ---------- --------- ---------
0                         0          1         1
```

8. Since we see that acorn2 has affinity for FTAG-1, it will forward the stream to the downstream receiver. Acorn1 will route the packet and forward it to any orphan ports, but drop the packets towards the vPC interface. The affinity forwarding decision is only relevant for vPC hosts, and does not apply to orphan ports. We saw previously that acorn1 not only had the vPC member receiver, but also had an orphan receiver on port channel 9.

```
acorn1%PineForest:Tree1# show fabricpath mroute vni 16100

(vlan/16100, 0.0.0.0, 225.1.110.112), uptime: 1d07h, igmp
 Outgoing interface list: (count: 2)
  Interface port-channel9, uptime: 00:19:16, igmp
  Interface port-channel10, uptime: 1d07h, igmp
```

9. Finally, verify that the multicast entry has properly been pushed down to the hardware forwarding ASICs as all packet forwarding is done in hardware. If an entry is not in hardware, then the result would not be intermittent traffic loss, but instead complete traffic loss for local receivers.

```
acorn1%PineForest:Tree1# show system internal forwarding vrf pineForest:Tree1 multicast route group
225.1.110.112

Multicast routes for table PineForest:Tree1/base
Dev | Index     | Group              | Source             | MET Ptr | RPF Int/
    |           |                    |                    |         | B-RP ord
----+-----------+--------------------+--------------------+---------+----------
1    0x148b      225.1.110.112/32     10.1.2.110/32        0x2       0x7a
```

You have now successfully verified multicast data-plane forwarding from a source to a receiver, and can use this information to follow the path of the packet and isolate any data-plane forwarding issues if troubleshooting is needed.

6 Cheat Sheets

The Cheat Sheets section is designed to provide you with a quick snapshot of show tech-support commands to gather before trying to recover from or clear any issue you are troubleshooting. Typically, it is best to gather the outputs from as many devices as possible that may be related to the issue. For example, if you are troubleshooting an auto-configuration issue on a vPC+ pair, gather the commands from both devices. For any forwarding issues, it is best to gather output logs from the source and destination device, as well as the spine switches/BGP route reflectors, and border leaf switches.

6.1 Profile Instantiation Cheat Sheet

Control-Plane Verification	Useful Commands	Information
show tech detail	For TAC use	Provides overall system-wide information
show tech port-profile	**show system internal config-profile history** **show system internal config-profile applied-config database** **show system internal config-profile config database**	Provides information on config-profile attempts, successes, and failures
show tech session-mgr	**show ip arp detail** **show ip arp internal event-history event**	Provides relevant timestamp and statistics for session manager events
debug port-profile all	**show fabric database host detail** **show fabric forwarding internal event-history auto-config**	Provides debug information for all port-profile transactions

6.2 HMM Cheat Sheet

Control-Plane Verification	Useful Commands	Information
show tech detail	For TAC use	Provides overall system-wide information
show tech auto-config	**show fabric forwarding ip local-host-db vrf all** **show system internal config-profile history**	Combines commands from evb, ecp, and fabric forwarding tables
show tech arp	**show ip arp detail** **show ip arp internal event-history event**	ARP events and tables for HMM entries and profile instantiation
show tech fabric forwarding	**show fabric database host detail** **show fabric forwarding internal event-history auto-config**	Provides relevant data from HMM process, such as profile instantiation and event-histories
show tech evb	**show evb vsi detail** **show evb hosts detail**	Provides relevant data from EVB process, such as VSI associations and event-histories
show tech ecp	**show ecp detail**	Provides relevant data from ECP process such as ECP transport statistics and event-histories

6.3 Unicast Forwarding Cheat Sheet

Control-Plane Verification	Useful Commands	Information
show tech detail	For TAC use	Provides overall system-wide information

Control-Plane Verification	Useful Commands	Information
show tech fabric forwarding	**show fabric forwarding ip local-host-db** **show fabric forwarding ip remote-host-db**	Provides information on local and remote learned host route entries
show tech fabricpath isis	**show fabricpath route switchid** *x* **show fabricpath isis database detail**	Provides relevant fabricpath isis adjacency and switch information
show tech fabricpath topology	**show fabricpath topology ftag** **show fabricpath topology detail**	Provides information on fabricpath ftag and tree information
show ip arp **show ip route**	**show ip arp** **show ip route**	Provides ARP and RIB information for host routes

6.4 Multicast Forwarding Cheat Sheet

Control-Plane Verification	Useful Commands	Information
show tech detail	For TAC use	Provides overall system-wide information
show tech ip multicast	**show ip igmp snoop groups** **show ip pim route** **show ip mroute** **show fabric multicast ipv4** *mroute | rp-grange | ssm-range*	Provides information on all components related to multicast.
Data-Plane Verification	Useful Commands	Information

show tech ip multicast	**show vlan id** *vlan* **vn-segment** **show ip mroute** **show fabric multicast ipv4 mroute**	Provides information on the control plane as well as data-plane packet forwarding.
show tech fabricpath isis **show tech fabricpath topology** **show tech m2rib**	**show fabricpath mroute vni** **show fabricpath topology ftag active** **show fabricpath isis trees multidest** *ftag* **show fabricpath load-balance multicast ftag-selected vlan** *vlan* **post-routed**	Provides datapath port selection information.
show tech forwarding l2 multicast **show tech forwarding l3 multicast**	**show system internal forwarding vrf** *name* **multicast route group** *mcast group ip*	Provides confirmation that hardware datapath is programmed by control-plane

7 Tips and Tricks

This section provides a few helpful tools outside of the standard show commands provided throughout this document to assist in troubleshooting.

7.1 Ethanalyzer

Ethanalyzer is an NX-OS built-in packet-capturing tool utilized to capture packets to or from the system CPU and help identify control-plane issues. For most practical purposes, network administrators can use this to determine whether a device is correctly sending/receiving the expected BGP Joins, IGMP Joins, RP Registration, and initial mroute state creation via IP packet reception and punt.

Detailed Ethanalyzer use-case examples can be found at:

Ethanalyzer on Nexus 7000 Troubleshooting Guide

7.2 ELAM

Embedded Logic Analyzer Module (ELAM) is another built-in NXOS packet-capturing tool. While Ethanalyzer is best used to verify packet reception at the CPU level, ELAM is more for packet reception verification at the port-ASIC (data-plane) level. ELAM is an extremely efficient packet-capturing tool when there is a need to determine if a packet is reaching a specific device in the DFA fabric (though ELAM is not DFA specific). There is no need for an external sniffer, and the packet reception verification can be done right on the device of interest. Though packets forwarded through a DFA fabric topology are encapsulated, basic IP source and destination triggers can be utilized to capture most packets. Detailed ELAM use-case examples can be found at ELAM Overview. Given below are a few basic example triggers that can be used to capture key packets in the data-plane. Also given below are use-case examples to verify packet reception, for Cisco Nexus 5000/6000 and 7000 Series switches. The below examples can be used to verify packet reception on server facing ports or fabric facing core ports.

Cisco Nexus 5000/6000 Series ingress server facing port:

```
acorn3%PineForest:Tree1# elam slot all
acorn3%PineForest:Tree1(bigsur-elam)# trigger lu ingress ipv4 if source-ipv4-address_ipv4 10.1.2.109
destination-ipv4-address_ipv4 225.1.23.111
acorn3%PineForest:Tree1(bigsur-elam)# start capture
acorn3%PineForest:Tree1(bigsur-elam)# show capture lu
Ingress Interface: Ethernet1/10 IS PC
+----------------------------------------+
|              Lookup Vector             |
+--------------------+-------------------+
|       Field        |      Raw Value    |
+--------------------+-------------------+
---- snip ----
| CE_DA              | 0x01005e01176f       |
| CE_SA              | 0x000001120112       |
| CE_Q0_VLAN         | 112                  |
| L3_SA              | 10.1.2.109           |
| L3_DA              | 225.1.23.111         |
```

Cisco Nexus 5000/6000 Series ingress fabric core facing port:

```
acorn1%PineForest:Tree1(bigsur-elam)# show capture lu
Ingress Interface: Ethernet1/2 IS NOT A PC
+----------------------------------------+
|              Lookup Vector             |
+--------------------+-------------------+
|       Field        |      Raw Value    |
+--------------------+-------------------+
---- snip ----
| CDCE_DTAG_ETYPE    | 0x8903             |   <<<< Fabricpath encapsulation etype
| CE_Q0_VLAN         | 12                 |
| CE_Q1_VLAN         | 848                |   <<<< Q0+Q1 tags making up vni
| L3_SA              | 10.1.2.109         |
| L3_DA              | 225.1.23.111       |

Nexus 7700 (Ingress on Spine switch):

Oak2# attach mod 1
module-1# elam asic flanker instance 0
```

```
module-1(fln-elam)# layer2
module-1(fln-l2-elam)# trigger dbus ipv4 ingress if source-ipv4-address 10.1.2.109 destination-ipv4-
address 225.1.23.111
module-1(fln-l2-elam)# trigger rbus ingress if trig
module-1(fln-l2-elam)# status
ELAM Slot 1 instance 0: L2 DBUS Configuration: trigger dbus ipv4 ingress if source-ipv4-address
10.1.2.109 destination-ipv4-address 225.1.23.111
L2 DBUS: Triggered      <<<<< triggered means we have received the packet.
ELAM Slot 1 instance 0: L2 RBUS Configuration: trigger rbus ingress if trig
L2 RBUS: Triggered

module-1(fln-l2-elam)# show dbus
---- snip ----
dtag-ftag           : 0x1             valid               : 0x1
source-ipv4-address: 10.1.2.109
destination-ipv4-address: 225.1.23.111      <<<< verify expected packet
```

8 Glossary

ADBM (Asset Database Manager): stores pre-defined network parameters

Backbone VLAN: Used for MP-BGP neighbor sessions to exchange the host-route information of the Cisco DFA enhanced control plane

BD (Bridge Domain): Layer 2 broadcast domain

BDI (Bridge Domain Interface): Layer 3 interface associated with BD (aka SVI)

BGP (*,G) and (S,G) Join: Used in place of PIM Joins through the Fabric

BL (Border Leaf switch): Provides L3 connectivity in and out of the Fabric

CFS (Cisco Fabric Services): Provides automatic configuration synchronization between devices.

DCNM (Data Center Network Manager) Provides the management of Datacenter into a single dashboard.

ESWID (Emulated Switch-ID): Switch-ID used in vPC+ pair

Fabric Forwarder: In dual border-leaf, used to elect a single forwarder of multicast traffic

HMM (Host Mobility Manager): Detects and tracks the movement of end hosts via ARP, NDP, or VDP.

L2FM (Layer 2 Feature Manager): Process that manages mac address table

L3 Core Dynamic Vlan: Used for the VRF VLAN-to-Segment-ID mapping

Layer2-VNI (segment-id): Globally significant ID that maps to a Leaf switch's 802.1q tagged vlan. Essentially what defines a broadcast domain across a Fabric.

Layer3-VNI (segment-id): Globally significant ID that maps to a tenant VRF

RD (Route Distinguisher): BGP route identifier made up of AS+L3 VNI

Server Leaf: Leaf switch with hosts directly attached

SWID: Fabricpath Switch-ID

VDP (VSI Discovery and Configuration Protocol): Communicates the presence of end-host Virtual Machines (VMs) and respective parameters to adjacent leaf nodes

VN-segment (VNI): New way to "tag" packets on the wire replacing the traditional 802.1Q VLAN tag with a 24-bit tag (allowing for higher vlan scale)

9 Related Links

- Cisco Dynamic Fabric Automation Technology: Overview and Deployment Considerations
- Automate Provisioning of Multitenant Cloud Environments with Cisco Dynamic Fabric Automation
- Simplified Fabric Management with Cisco Dynamic Fabric Automation
- Cisco Dynamic Fabric Automation Configuration Guide
- Cisco Dynamic Fabric Automation Troubleshooting Guide
- Cisco Dynamic Fabric Automation White Papers