Troubleshooting ACI

(DCACIO v4.0)

Agenda

- Troubleshooting Checklist
- Troubleshooting APIC
- Faults in ACI
- Visore
- Lab07 - Visore
- CLI Troubleshooting Commands
- Lab08 - Moquery
- Lab09 - iPing / iTraceroute
- Troubleshooting Endpoint Connectivity
- Endpoint Move Scenarios
- Visibility & Troubleshooting Tool
- Troubleshooting Access Ports
- Troubleshooting the Fabric Discovery Process (LLDP)
- Hardware Diagnostics and Replacement

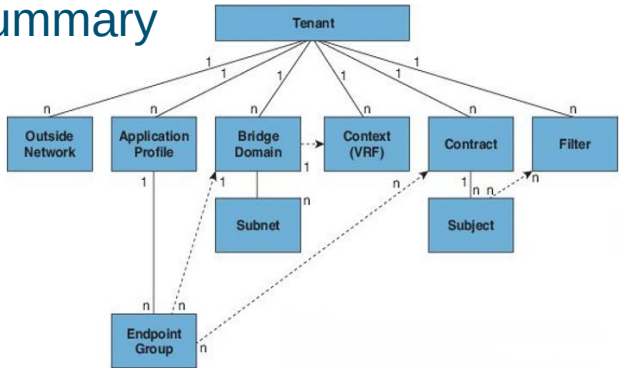
Troubleshooting Checklist

(DCACIO v4.0)

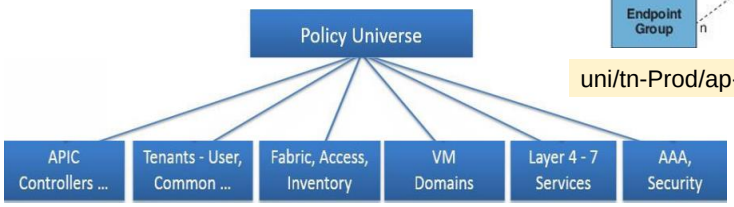
Object Model Visual Summary

Everything in ACI is an Object

- Distinguished Names (DN)
- Managed Object (MO) Types:
  - **Logical**: user configurable
  - **Resolved**: system generated
  - **Concrete**: coded into systems



uni/tn-Prod/ap-CatVideos/epg-WEB/rscons-Intake



Logical model, Resolved model, concrete model

Within the ACI object model, there are essentially three stages of implementation of the model: the Logical Model, the Resolved Model, and the Concrete Model.

- The **Logical Model** is the logical representation of the objects and their relationships. The AP that was discussed previously is an expression of the logical model. This is the declaration of the "end-state" expression that is desired when the elements of the application are connected and the fabric is provisioned by the APIC, stated in high-level terms.
- The **Resolved Model** is the abstract model expression that the APIC resolves from the logical model. This is essentially the elemental configuration components that would be delivered to the physical infrastructure when the policy must be executed (such as when an endpoint connects to a leaf).
- The **Concrete Model** is the actual in-state configuration delivered to each individual fabric member based on the resolved model and the Endpoints attached to the fabric.

In general, the logical model should be the high-level expression of what exists in the resolved model, which should be present on the concrete devices as the concrete model expression. If there is any gap in these, there will be inconsistent configurations.

Troubleshooting Checklist

- ✓ **Check Health scores** – narrow down affected scope
- ✓ **Check Faults** – verify deployment failures
- ✓ **Check Audit Log** – verify changes to the configuration
- ✓ **Check Resolved Object Model** – is it present in both APIC and relevant leaf switches?
- ✓ **Check Concrete Objects** – are they present on the relevant leaf switches?
- ✓ **Verify using CLI** – SSH to APIC and leaf switches

Troubleshooting ACI goes through some specific problem descriptions as it relates to the fabric.

For each iterative problem, there will be a problem description, a listing of the process, some verification steps, and possible resolutions.

- **Problem Description**: The problem description can be a high level observation of the starting point for the troubleshooting actions to be covered. Example: a fabric node is showing “inactive” from the APIC by using APIC CLI command “acidiag fnvread”.
- **Symptoms**: Depending on the problem, various symptoms and their impacts may be observed. In this example, some of the symptoms and indications of issues around an inactive fabric node could be:
  - loss of connectivity to the fabric
  - low health score
  - system faults
  - inability to make changes through the APIC
- **Verification and cause**: The logical set of steps to identify what is being observed will be indicated along with the appropriate tools and output. Additionally, some information about what is being observed and the likely causes will be included.

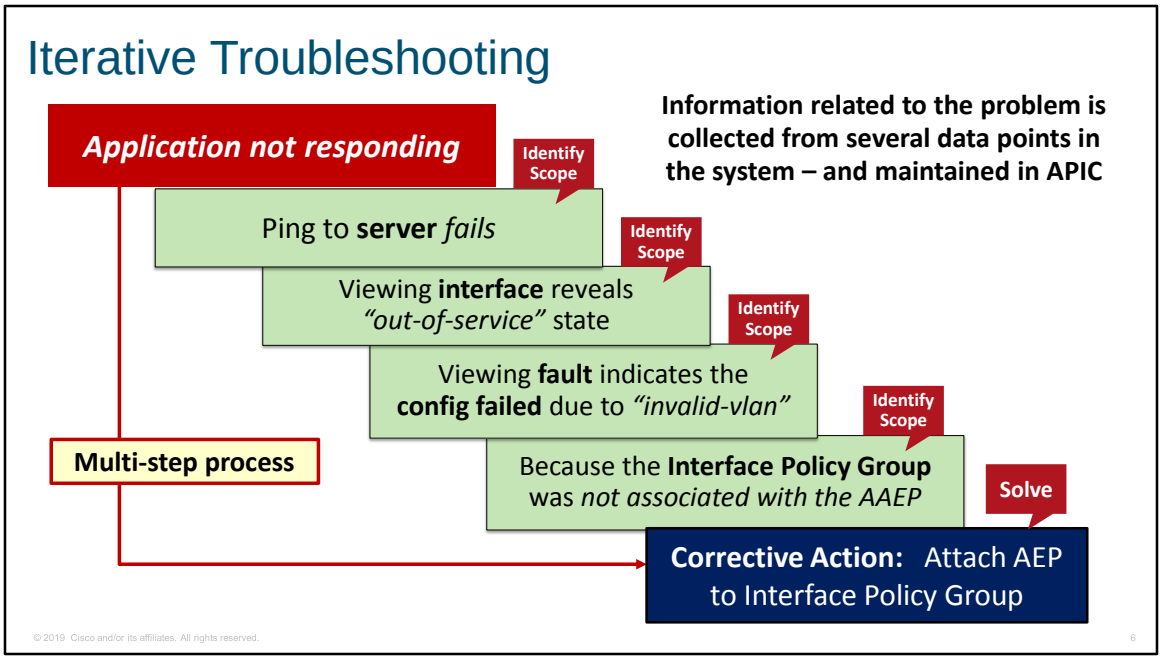
Troubleshooting Checklist -

- Check Health scores – narrow down affected scope
- Check Faults – faults raised if something fails during deployment
- Check if resolved Object Model is present in both APIC and relevant leaf switches
- Check if the Concrete Objects are present on the relevant leaf switches
- Verify using CLI – SSH to leaf switches; invoke iNXOS shell commands

Visore, the Application Policy Infrastructure Controller (APIC) Object Store Browser. You can use it in order to directly query the Managed Objects (MO) when you point your browser to either the IP address of one of the APICs or a physical switch. One would typically point to the APIC which is generally HTTPS (default), but could also be HTTP if configured.

https://<APIC or Switch IP ADDRESS>/visore.html

Using the more traditional NX-OS style commands in the APIC CLI can present specific information in a text format. In addition, administrators can query the APIC object database directly using the CLI command "**moquery**". This command is referred to as the command-line cousin to Visore. Moquery requires you first SSH to the APIC. More on moquery command options are presented later in this lesson.



Troubleshooting Methodology

Troubleshooting is the systematic process used to identify the cause of a problem. The problem to be addressed is determined by the difference between how some entity (function, process, feature, etc.) should be working versus how it is working. Once the cause is identified, the appropriate actions can be taken to either correct the issue or mitigate the effects: the latter is sometimes referred to as a workaround.

Troubleshooting is an iterative process attempting to isolate an issue to the point that some action can be taken to have a positive effect. Often this is a multi-step process which moves toward isolating the issue. For example, in deploying an application on a server attached to an ACI fabric, a possible problem observed could be that the application does not seem to respond from a client on the network. The isolation steps may look something like the diagram.

Troubleshooting is usually not a simple linear path, and in this example it is possible that a troubleshooter may have observed the system fault earlier in the process and started at that stage.

In this example, information related to the problem came from several data points in

the system. These data points can be part of a linear causal process or can be used to better understand the scope and various points and conditions that better define the issue. How these data points are collected is defined by three characteristics:

- WHAT: What information is being collected
- WHERE: Where on the system is the information being collected
- HOW: The method used in collecting the information

For example, the state of the fabric Ethernet interface can be gathered from the leaf through CLI on the leaf in a couple of different ways. This information can be gathered from the APIC, either through the GUI or the REST API call. When troubleshooting, it is important to understand where else relevant information is likely to come from to build a better picture of what is the issue.

Troubleshooting APIC

(DCACIO v4.0)

Startup Script

The only way to change key values is rebuild fabric

- **Fabric Name**
- **TEP address**
- **Infra VLAN**

TEP should be unique

- Default: 10.0.0.0/16
- /16 is safe if future scale is uncertain
- /22 smallest supported pre 2.0
- /23 smallest supported post 2.0

```
Cluster configuration ...
→ Fabric name: Fabric-1
  Fabric ID: 1
  Number of controllers: 1
  Controller name: APIC-1
  POD ID: 1
  Controller ID: 1
→ TEP address pool: 10.0.0.0/16
→ Infra VLAN ID: 3914
  Multicast address pool: 225.0.0.0/15

Out-of-band management configuration ...
  Management IP address: 10.48.25.77
  Default gateway: None
  Interface speed/duplex mode: auto

admin user configuration ...
  Strong Passwords: N
  User name: admin
  Password: ********

The above configuration will be applied ...

Warning: TEP address pool, Infra VLAN ID and Multicast address
pool cannot be changed later, these are permanent until the
fabric is wiped.

Would you like to edit the configuration? (y/n) [n]:
```


Three key settings in the APIC initial setup cannot be modified after implementation:

- Fabric Name
- TEP address
- Infra VLAN

The only method to modify any of these settings is to wipe all APIC controllers and perform the initial setup again for each APIC controller.

- This forces a new fabric discovery of the switch nodes

Cisco TAC recommends the VXLAN Tunnel Endpoint (TEP) IP address be unique system-wide; no overlap with any other device on the network. The default TEP IP address designated in the initial setup menu is 10.0.0.0/16. The latest Cisco ACI code release supports a CIDR mask of /23. To allow for the most possible host addresses, Cisco TAC recommends /16; especially if it is unknown the scale the ACI fabric may grow over time.

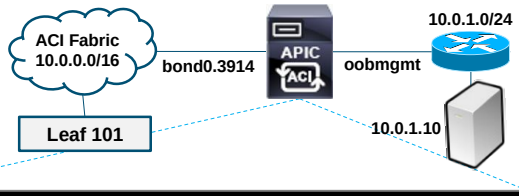
Why Unique TEP Range?

APIC has no concept of VRFs

- Given the routing table, return traffic sourced from the APIC within the TEP defined pool will go towards **infra**
- Extends to **inbmgmt** as well as the docker subnet, both not shown here
- Notice the **GW for the 10.0.0.0/16 route pointing to 10.0.0.30** (infra)

APIC Routes

- Default route and oobmgmt specific subnet points to oobmgmt interface

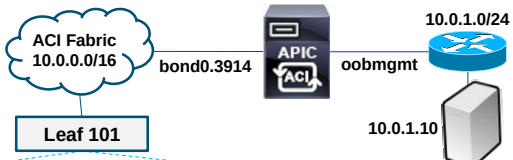



```
apic1> ifconfig oobmgmt
oobmgmt: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.48.25.77  netmask 255.255.255.0  broadcast 10.48.25.255

apic1> ifconfig bond0.3914
bond0.3914: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1496
        inet 10.0.0.1  netmask 255.255.255.255  broadcast 10.0.0.1

apic1> route
Kernel IP routing table
Destination  Gateway        Genmask          Flags Metric Ref  Use  Iface
0.0.0.0      10.48.25.100   0.0.0.0          UG    16     0      0  oobmgmt
10.0.0.0     10.0.0.30      255.255.0.0      UG    0      0      0  bond0.3914
10.0.0.30    0.0.0.0        255.255.255.255  UH    0      0      0  bond0.3914
10.0.8.65    10.0.0.30      255.255.255.255  UGH   0      0      0  bond0.3914
10.0.8.66    10.0.0.30      255.255.255.255  UGH   0      0      0  bond0.3914
10.48.25.0   0.0.0.0        255.255.255.0    U     0      0      0  oobmgmt
```


The reason for a unique, network-wide TEP IP address is the APIC has no virtual routing and forwarding (VRF) concept. No initial setup setting designates an APIC VRF.

In the diagram, IP address **10.48.25.100** on the router indicates is it "default gateway" for the **oobmgmt** interface on APIC. The diagram illustrates a circumstance where an overlapping IP address range exists on the router. In this example, a syslog or AD server with IP address 10.0.1.10 would not be reachable from APIC because of the TEP range overlap.

Why Unique TEP Range?



Here 10.0.0.30/32 is acting as GW for the APIC route

- All assigned TEPs are /32s which stem from the startup script defined TEP pool
- Note that it is mapping access encap 3914 to platform independent VLAN 7

```
leaf101# show ip int vrf overlay-1
vlan7, Interface status: protocol-up/link-up/admin-up, iod: 65, mode: unspecified
  IP address: 10.0.0.30, IP subnet: 10.0.0.0/27
  IP broadcast address: 255.255.255.255
  IP primary address route-preference: 1, tag: 0

module-1# show system internal eltmc info vlan brief
VLAN-Info
VlanId  HW_VlanId Type          Access_enc Access_enc Fabric_enc Fabric_enc  BDVlan
                               Type                  Type
================================================================================
7       11        BD_CTRL_VLAN  802.1q     3914       VXLAN      16777209    0
```


In the diagram IP address 10.0.0.30/32 is acting as the gateway for the APIC route.

All assigned Tunnel; end-points (TEPs) are /32 subnets, which stem from the startup script defined TEP pool.

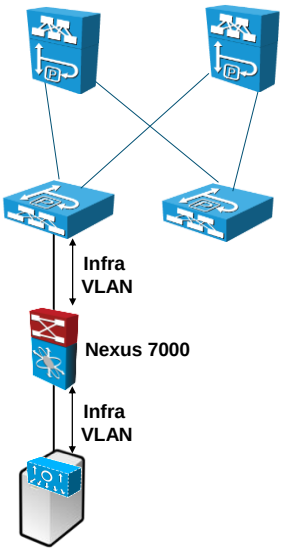
Note: that it is mapping access encap 3914 to platform independent vlan 7

The Infra VLAN

Defined once within the startup script – cannot change afterwards

- The Infra VLAN should be an unused VLAN relative to your datacenter
- It may be necessary to trunk the Infra VLAN to support AVS, AVE or any other opflex integrations
- Common reserved range is **3915 to 4095**
- Choose a VLAN that is not normally reserved
  - e.g. 3914

Note: VXLAN must be sent on the infra VLAN for ACI to be able to parse it



As previously mentioned, the Infra VLAN is defined once within the startup script and cannot be modified change afterwards with total fabric disruption. A complete rebuild and fabric discovery is required.
The requires the Infra VLAN be unique across the datacenter.
Certain circumstances dictate this requirement, for example:

- Trunking the Infra VLAN to support AVS, AVE or any other opflex integrations
- On certain Cisco platforms (example: Cisco UCS) there are common reserved VLAN ranges: 3915 to 4095. Choose a VLAN that is not normally reserved – e.g. 3914

APIC & ACI – A Crypto Based Platform

User and Orchestration access to APIC

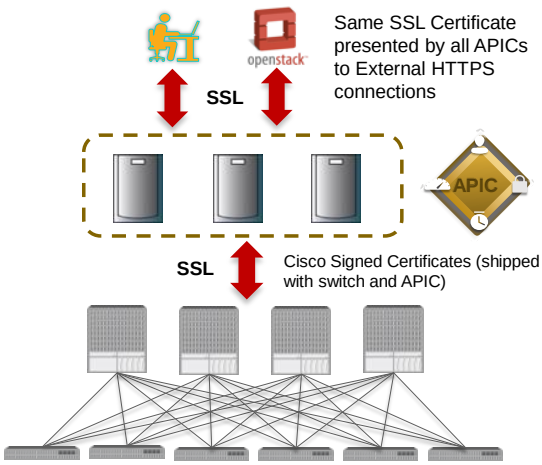
- Web-Token or X.509 based certs
- APIC to Switch - SSL connection leveraging public key certificates

APIC ISO is encrypted

- Certs are stored on APIC TPM and Keys to access TPM are stored within a **USB key inside the APIC**

luks or **tpm** errors booting APIC may indicate:

- TPM ownership has been disabled
- Certs have gone bad or were cleared
- Someone removed the USB from the APIC



APIC and the ACI fabric are a crypto-based platform. User and Orchestration access to APIC employ a Web-Token or X.509 based certificates. SSL connection leveraging public key certificates is implemented between the APIC to all switch nodes.

- The APIC ISO is encrypted
- Certificates are stored on APIC TPM and Keys to access TPM are stored within a **USB key inside the APIC**

During APIC bootup, certain luks or tpm errors may indicate:

- TPM ownership has been disabled
- Certs have gone bad or were cleared
- Someone removed the USB from the APIC

APIC Cluster Size

Operation to increase any size mismatches, from a cluster size of N to size N+1

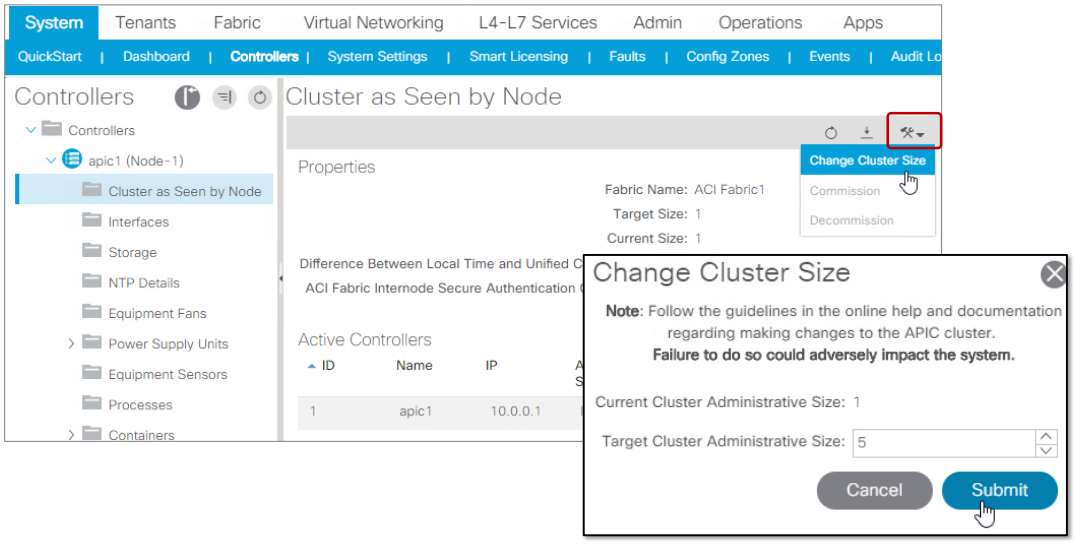
- Increase the cluster target size to be equal to the existing cluster size controller count plus the new controller count
- Depending on the amount of data the APIC must synchronize upon the addition of each appliance, the time required to complete the expansion could be ***more than 10 minutes*** per appliance
- If one or more of the APIC controllers' health status in the cluster is not "fully fit", remedy that situation before proceeding

The **APIC cluster** is comprised of multiple APIC controllers that provide operators a unified real time monitoring, diagnostic, and configuration management capability for the ACI fabric. To assure optimal system performance, follow the guidelines below for making changes to the APIC cluster. Note - prior to initiating a change to the cluster, always verify its health. When performing planned changes to the cluster, all controllers in the cluster should be healthy. If one or more of the APIC controllers' health status in the cluster is not "fully fit", remedy that situation before proceeding. See the Cisco APIC Troubleshooting Guide for more information on resolving APIC cluster health issues.

Follow these general guidelines when managing clusters:

- Disregard cluster information from APICs that are not currently in the cluster; they do not provide accurate cluster information.
- Cluster slots contain an APIC Chassis ID. Once you configure a slot, it remains unavailable until you decommission the APIC with the assigned Chassis ID.
- If an APIC firmware upgrade is in progress, wait for it to complete and the cluster to be fully fit before proceeding with any other changes to the cluster.

Expanding APIC Cluster Size



Expanding the Cisco APIC cluster is the operation to increase any size mismatches, from a cluster size of N to size N+1, within legal boundaries. The operator sets the administrative cluster size and connects the APICs with the appropriate cluster IDs, and the cluster performs the expansion.
During cluster expansion, regardless of in which order you physically connect the APIC controllers, the discovery and expansion takes place sequentially based on the APIC ID numbers. For example, APIC2 is discovered after APIC1, and APIC3 is discovered after APIC2 and so on until you add all the desired APICs to the cluster. As each sequential APIC is discovered, a single data path or multiple data paths are established, and all the switches along the path join the fabric. The expansion process continues until the operational cluster size reaches the equivalent of the administrative cluster size.

Expanding the Cluster Size – follow these **guidelines** to expand the APIC cluster size:

- Schedule the cluster expansion at a time when the demands of the fabric workload will not be impacted by the cluster expansion.
- If one or more of the APIC controllers' health status in the cluster is not "fully fit", remedy that situation before proceeding.
- Stage the new APIC controller(s) according to the instructions in their hardware

installation guide. Verify in-band connectivity with a PING test.

- Increase the cluster target size to be equal to the existing cluster size controller count plus the new controller count. For example, if the existing cluster size controller count is 3 and you are adding 3 controllers, set the new cluster target size to 6. The cluster proceeds to sequentially increase its size one controller at a time until all new the controllers are included in the cluster. Note: Cluster expansion stops if an existing APIC controller becomes unavailable. Resolve this issue before attempting to proceed with the cluster expansion.
- Depending on the amount of data the APIC must synchronize upon the addition of each appliance, the time required to complete the expansion could be more than 10 minutes per appliance. Upon successful expansion of the cluster, the APIC operational size and the target size will be equal. Note: Allow the APIC to complete the cluster expansion before making additional changes to the cluster.

Expanding the APIC Cluster Using the GUI – Procedure:

Step 1 On the menu bar, choose SYSTEM > Controllers. In the Navigation pane, expand Controllers > apic_controller_name > Cluster.

You must choose an apic_controller_name that is within the cluster that you wish to expand. In the Work pane, the cluster details are displayed. This includes the current cluster target and current sizes, the administrative, operational, and health states of each controller in the cluster.

Step 2 Verify that the health state of the cluster is Fully Fit before you proceed with contracting the cluster.

Step 3 In the Work pane, click Actions > Change Cluster Size.

Step 4 In the Change Cluster Size dialog box, in the Target Cluster Administrative Size field, choose the target number to which you want to expand the cluster. Click Submit.

Note: It is not acceptable to have a cluster size of two APIC controllers. A cluster of one, three, or more APIC controllers is acceptable.

Step 5 In the Confirmation dialog box, click Yes.

In the Work pane, under Properties, the Target Size field must display your target cluster size.

Step 6 Physically connect all the APIC controllers that are being added to the cluster.

In the Work pane, in the Cluster > Controllers area, the APIC controllers are added one by one and displayed in the sequential order starting with N + 1 and continuing until the target cluster size is achieved.

Step 7 Verify that the APIC controllers are in operational state, and the health state of each controller is Fully Fit.

Expanding the APIC Cluster Using the REST API

Step 1 Set the target cluster size to expand the APIC cluster size.

Example:

POST

```
https://<IP address>/api/node/mo/uni/controller.xml
<infraClusterPol name='default' size=3/>
```

Step 2 Physically connect the APIC controllers that you want to add to the cluster.

Contracting APIC Cluster

Reducing the cluster size increases the load on the remaining APIC controllers

- **Task sequence** – one by one until cluster reaches the lower target size:
  - **Decommission APIC controller**
    - Starting with highest numbered controller ID in existing cluster
  - **Power down**
  - **Disconnect the APIC controller**
- Failure to follow an orderly process to decommission and power down APIC controllers from a reduced cluster can lead to unpredictable outcome
- **Note**: Cluster synchronization stops if an existing APIC controller becomes unavailable

Contracting the Cisco APIC cluster is the operation to decrease any size mismatches, from a cluster size of N to size N -1, within legal boundaries. As the contraction results in increased computational and memory load for the remaining APICs in the cluster, the decommissioned APIC cluster slot becomes unavailable by operator input only.
During cluster contraction, you must begin decommissioning the last APIC in the cluster first and work your way sequentially in reverse order. For example, APIC4 must be decommissioned before APIC3, and APIC3 must be decommissioned before APIC2.

Contracting the Cluster Size – follow the **guidelines** below to reduce the APIC cluster size, decommission the APIC controller appliances that are removed from the cluster. Failure to follow an orderly process to decommission and power down APIC controller appliances from a reduced cluster can lead to unpredictable outcomes. Do not allow unrecognized APIC controller appliances to remain connected to the fabric. Reducing the cluster size increases the load on the remaining Note APIC controller appliances.

- Schedule the APIC controller size reduction at a time when the demands of the fabric workload will not be impacted by the cluster synchronization.
- Reduce the cluster target size to the new lower value. For example if the existing

cluster size is 6 and you will remove 3 controllers, reduce the cluster target size to 3.

- Starting with the highest numbered controller appliance ID in the existing cluster, decommission, power down, and disconnect the APIC controller appliance that will be replaced one by one until the new lower target number is reached. Upon the decommissioning and removal of each controller appliance, the APIC synchronizes the cluster.
- Cluster synchronization stops if an existing APIC appliance becomes unavailable. Resolve this issue before attempting to proceed with the cluster synchronization.
- Depending on the amount of data the APIC must synchronize upon the replacement of an appliance, the time required to complete decommissioning each controller appliance could be more than 10 minutes per appliance.

Contracting the APIC Cluster Using the REST API

The cluster drives its actual size to the target size. If the target size is lower than the actual size, the cluster size contracts.

Procedure

Step 1 Set the target cluster size so as to contract the APIC cluster size.

Example:

POST

```
https://<IP address>/api/node/mo/uni/controller.xml
<infraClusterPol name='default' size=1/>
```

Step 2 Decommission APIC3 on APIC1 for cluster contraction.

Example:

POST

```
https://<IP address>/api/node/mo/topology/pod-1/node-1/av.xml
<infraWiNode id=3 adminSt='out-of-service'/>
```

Step 3 Decommission APIC2 on APIC1 for cluster contraction.

Example:

POST

```
https://<IP address>/api/node/mo/topology/pod-1/node-1/av.xml
<infraWiNode id=2 adminSt='out-of-service'/>
```

Faults in ACI

(DCACIO v4.0)

Fault Overview

A fault is a Managed Object (MO) contained in M.I.T.

- A MO is automatically created, escalated, de-escalated, and deleted by the system as specific conditions are detected
- **Severity:** Critical, Major, Minor, Warning, Info, Cleared
- **Domains**: Access, External, Framework, Infra, Management, Security, Tenant
- **Types**: Generic, Equipment, Configuration, Connectivity, Environmental, Management, Network

Fault Counts By Domain

Fault Level:				
SYSTEM WIDE	1	6	96	19
Access	0	0	0	16
External	0	0	0	0
Framework	0	0	0	0
Infra	1	6	24	3
Management	0	0	0	0
Security	0	0	0	0
Tenant	0	0	72	0

Fault Counts By Type

Fault Level:				
Communications	0	0	0	16
Config	0	0	82	3
Environmental	0	0	0	0
Operational	1	6	14	0

About Faults

The Application Policy Infrastructure Controller (APIC) maintains a comprehensive, up-to-date run-time representation of the administrative and operational state of the Cisco Application Centric Infrastructure (ACI) fabric system in a collection of managed objects (MOs). In this model, a fault is represented as a mutable, stateful, and persistent MO. When a specific condition occurs, such as a component failure or an alarm, the system creates a fault MO as a child object to the MO that is primarily associated with the fault. For a fault object class, the fault conditions are defined by the fault rules of the parent object class.

In most cases, **a fault MO is automatically created, escalated, de-escalated, and deleted by the system as specific conditions are detected**. If the same condition is detected multiple times while the corresponding fault MO is active, no additional instances of the fault MO are created. A fault MO remains in the system until the fault condition is cleared. The fault MO is deleted according to the settings in the fault collection and fault retention policies. A fault MO is read-only unless it is in the cleared and retained state, when it can be deleted by the user by acknowledging it.

The creation of a fault MO can be triggered by internal processes such as finite state

machine (FSM) transitions or detected component failures, or by conditions specified by various fault policies, some of which are user configurable. For example, you can set fault thresholds on statistical measurements such as health scores, data traffic, or temperatures.

Note: For detailed reference information about faults, events, errors, and system messages, see the Cisco ACI System Messages Reference Guide or the Cisco APIC Management Information Model Reference, which is a web-based application.

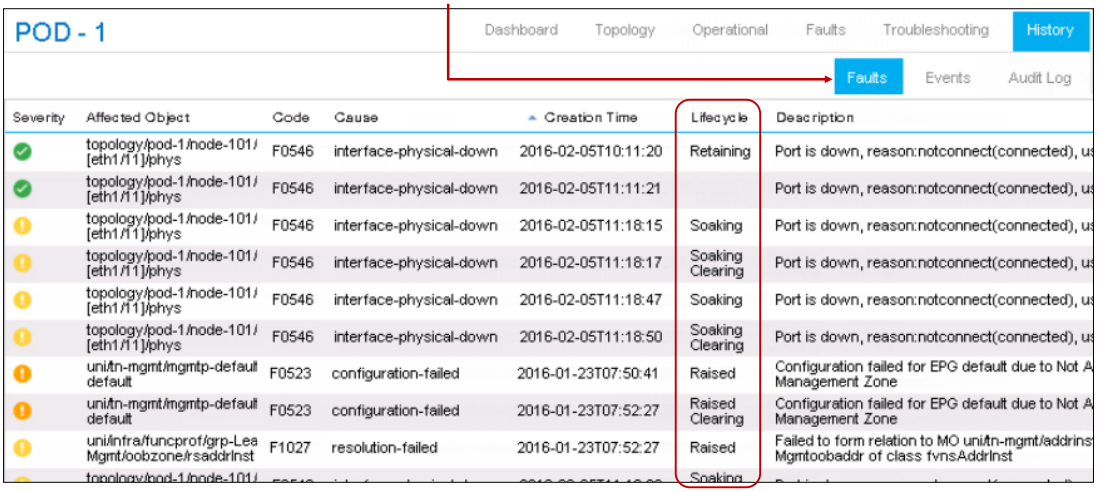
Fault Severity

A fault raised by the system can transition through more than one severity during its life cycle. This table describes the possible fault severities in decreasing order of severity

- **critical** - A service-affecting condition that requires immediate corrective action. For example, this severity could indicate that the managed object is out of service and its capability must be restored.
- **major** - A service-affecting condition that requires urgent corrective action. For example, this severity could indicate a severe degradation in the capability of the managed object and that its full capability must be restored.
- **minor** - A nonservice-affecting fault condition that requires corrective action to prevent a more serious fault from occurring. For example, this severity could indicate that the detected alarm condition is not currently degrading the capacity of the managed object.
- **warning** - A potential or impending service-affecting fault that currently has no significant effects in the system. An action should be taken to further diagnose, if necessary, and correct the problem to prevent it from becoming a more serious service-affecting fault.
- **info** - A basic notification or informational message that is possibly independently insignificant. (Used only for events)
- **cleared** - A notification that the condition that caused the fault has been resolved, and the fault has been cleared.

Viewing Faults

Drill-down via **Dashboard**; “Faults” tab as-needed per-context



Viewing Faults Using the GUI

To view a summary of fault statistics for the overall system, click SYSTEM in the menu bar of the APIC GUI and click FAULTS in the submenu bar. The fault counts by domain and by type are displayed in the dashboard tables.

Logged faults are presented in many places in the GUI, filtered to show only those faults that are relevant to the current GUI context. Wherever a Faults tab appears in the GUI Work pane, you can view the relevant entries from the fault log. This procedure shows how to view tenant faults as an example.

Procedure:

Step1 - In the menu bar, click Tenants.

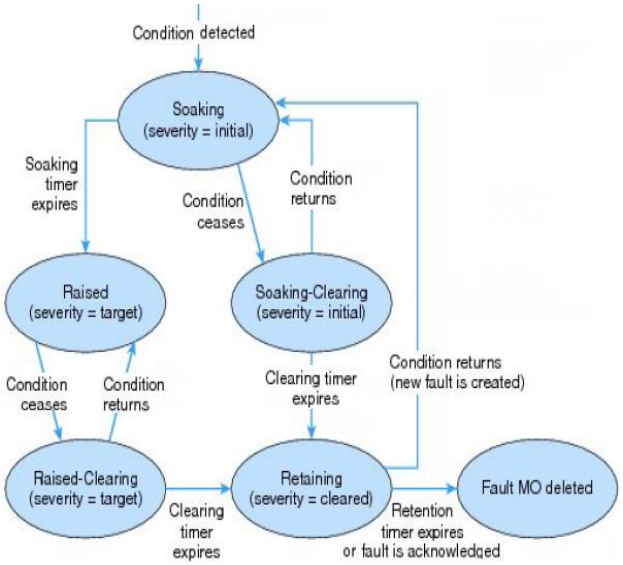
Step2 - In the submenu bar, click the name of the tenant.

Step3 - In the Navigation pane, click the name of the tenant.

Step4 - In the Work pane, click the Faults tab.

Fault Lifecycle

- **Soaking (2min):** 'fault MO' created when fault condition is detected
- **Soaking-Clearing** (**2min**): fault condition is alleviated during soaking interval
- **Raised**: fault condition persists when soaking interval expires
- **Raised-Clearing**: fault condition of a Raised Fault is alleviated
- **Retaining** (**1hr**): fault condition is absent for the duration of the clearing interval in either the **Raised-Clearing** or **Soaking-Clearing** state



Fault Life Cycle

APIC fault MOs are stateful, and a fault raised by the APIC transitions through more than one state during its life cycle. In addition, the severity of a fault might change due to its persistence over time, so a change in the state may also cause a change in severity. Each change of state causes the creation of a fault record and, if external reporting is configured, can generate a syslog or other external report.

Only one instance of a given fault MO can exist on each parent MO. If the same fault occurs again while the fault MO is active, the APIC increments the number of occurrences.

The fault life cycle is shown in the state diagram.

The characteristics of each state are as follows:

- **Soaking** —A fault MO is created when a fault condition is detected. The initial state is Soaking, and the initial severity is specified by the fault policy for the fault class. Because some faults are important only if they persist over a period of time, a soaking interval begins, as specified by the fault policy. During the soaking interval, the system observes whether the fault condition persists or whether it is alleviated and reoccurs one or more times. When the soaking interval expires, the next state depends on whether the fault condition remains.

- **Soaking-Clearing** — If the fault condition is alleviated during the soaking interval, the fault MO enters the Soaking-Clearing state, retaining its initial severity. A clearing interval begins. If the fault condition returns during the clearing interval, the fault MO returns to the Soaking state. If the fault condition does not return during the clearing interval, the fault MO enters the Retaining state.
- **Raised** —If the fault condition persists when the soaking interval expires, the fault MO enters the Raised state. Because a persistent fault might be more serious than a transient fault, the fault is assigned a new severity, the target severity. The target severity is specified by the fault policy for the fault class. The fault remains in the Raised state at the target severity until the fault condition is alleviated.
- **Raised-Clearing** —When the fault condition of a Raised Fault is alleviated, the fault MO enters the Raised-Clearing state. The severity remains at the target severity, and a clearing interval begins. If the fault condition returns during the clearing interval, the fault MO returns to the Raised state.
- **Retaining** —When the fault condition is absent for the duration of the clearing interval in either the Raised-Clearing or Soaking-Clearing state, the fault MO enters the Retaining state with the severity level cleared. A retention interval begins, during which the fault MO is retained for the length of time that is specified in the fault policy. This interval ensures that the fault reaches the attention of an administrator even if the condition that caused the fault has been alleviated, and that the fault is not deleted prematurely. If the fault condition reoccurs during the retention interval, a new fault MO is created in the Soaking state. If the fault condition has not returned before the retention interval expires, or if the fault is acknowledged by the user, the fault MO is deleted.

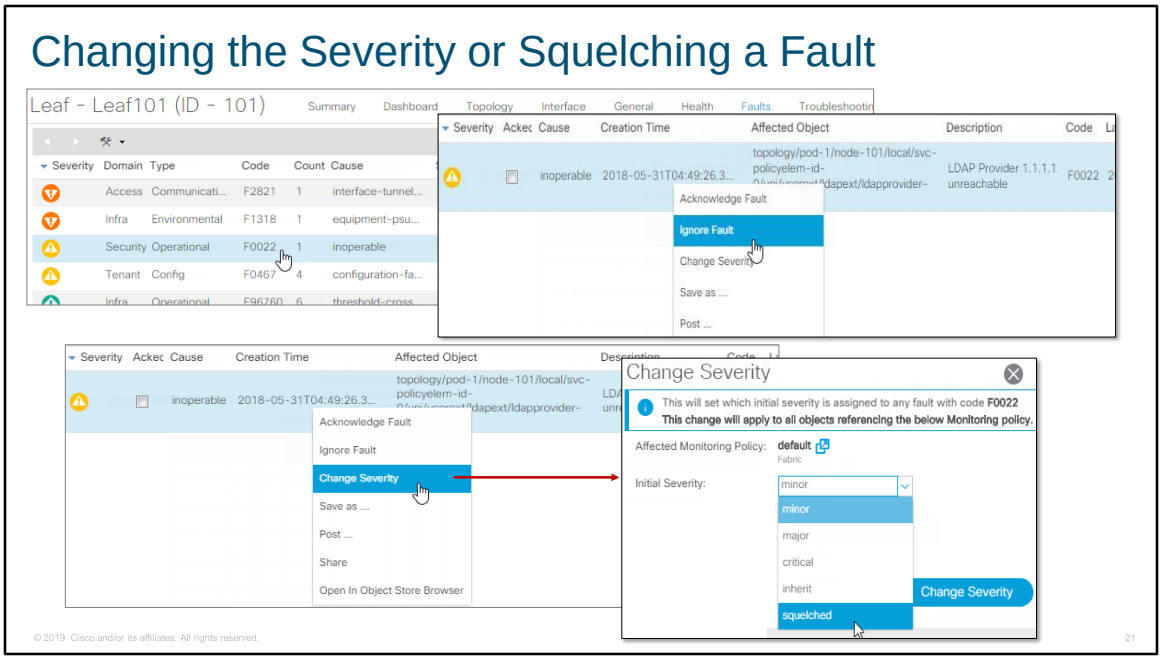
The soaking, clearing, and retention intervals are specified in the fault life cycle profile (fault:LcP) object.

Fault Types

Type	Description
generic	The system has detected a generic issue.
equipment	The system has detected that a physical component is inoperable or has another functional issue.
configuration	The system is unable to successfully configure a component.
connectivity	The system has detected a connectivity issue, such as an unreachable adapter.
environmental	The system has detected a power issue, thermal issue, voltage issue, or a loss of CMOS settings.
management	The system has detected a serious management issue, such as one of the following: • Critical services could not be started. • Components in the instance include incompatible firmware versions.
network	The system has detected a network issue, such as a link down.
operational	The system has detected an operational issue, such as a log capacity limit or a failed component discovery.

Fault Types

A fault raised by the system can be one of the types described in this table.



Changing the Severity or Squelching a Fault

Every APIC fault has a default severity. In some circumstances, a fault might be considered more or less severe than the default level. In some cases, you might want to ignore a particular fault and squelch (suppress) it from appearing in fault reports or status dashboards. APIC provides two locations from which you can change the severity of a fault type:

- Directly from a fault instance in the dashboard or fault table
- In a monitoring policy

Note: This feature was introduced in Cisco APIC Release 3.2(1) and modified in Cisco APIC Release 4.0(1).

Changing the Severity of a Fault in the Dashboard or Faults Table

In an APIC dashboard or fault table, you can change the severity of a displayed fault or you can suppress (squelch) it altogether. A few examples of APIC dashboard or fault table locations are as follows:

- System > Faults
- Tenants > mgmt > Tenant mgmt > Faults
- Fabric > Inventory > Pod 1 > Faults

- Fabric > Inventory > Pod 1 > leaf1 > Faults

Changing the Severity of a Fault in the Monitoring Policy
In a monitoring policy, you can change the severity of a fault or suppress (squelch) it altogether.

Viewing Faults in REST API and Visore

With REST API, query by class or by DN

- All faults in Fabric:

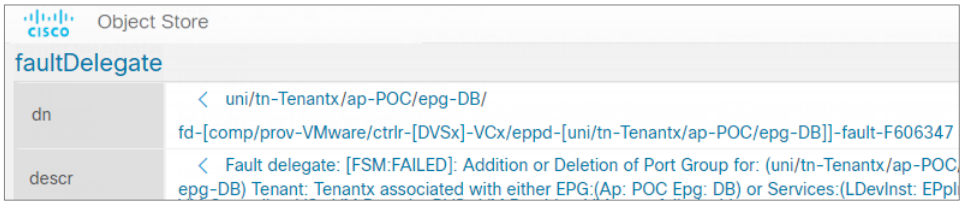
`https://apic1/api/node/class/faultInst.xml`

- Individual fault:

`https://apic1/api/node/mo/<fault-dn>.xml`

- All faults for MO:

`https://apic1/api/node/mo/<affected-dn>.xml?rsp-subtree-include=faults`



Viewing Faults Using the API

You can view faults using the API query methods to search for fault MOs, which can be of class fault:Inst or fault:Delegate. You can search for all instances or you can refine your search using query filters as described in the Cisco APIC REST API User Guide.

This example shows how to query a physical interface for the faults associated to it:
GET http://192.0.20.123/api/node/mo/topology/pod-1/node-1017/sys/phys-[eth1/11]/phys.xml?rsp-subtree-include=fault

This example shows how to request all the fault records associated to a multicast tree object:
GET http://192.0.20.123/api/node/mo/topology/pod-1/node-1017/sys/isis/inst-default/dom-overlay-1/fmtree-2.xml ?rsp-subtree-include=fault-records

Viewing Faults Using the NX-OS Style CLI

To display a summary of faults for a specific entity, enter the show faults command with the appropriate qualifiers. Some common forms of the show faults command

are the following:

- show faults
- show faults controller
- show faults leaf
- show faults leaf interface
- show faults spine
- show faults tenant

Viewing Faults Using the Object Model CLI

To display a summary of faults for a specific MO, navigate to the directory of that MO and enter the faults command. To display a summary of faults on a given node, module, port, or interface, enter the faults command with any of the following command forms:

- faults controller controller-number
- faults switch node-id
- faults switch interface interface-name
- faults switch module module-id
- faults switch module module-id port port-number
- faults path filepath

Additional options for these commands include the following:

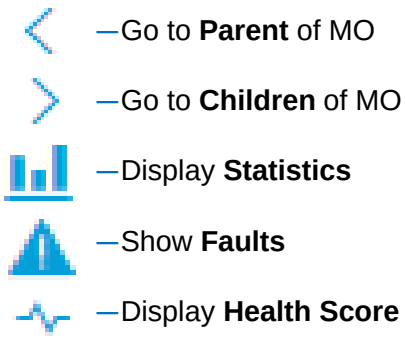
- detail —Displays fault detail.
- ack fault-code —Acknowledge a fault.
- unack fault-code —Unacknowledge a fault.
- history [fault-record-id] —Displays historical data for the fault.

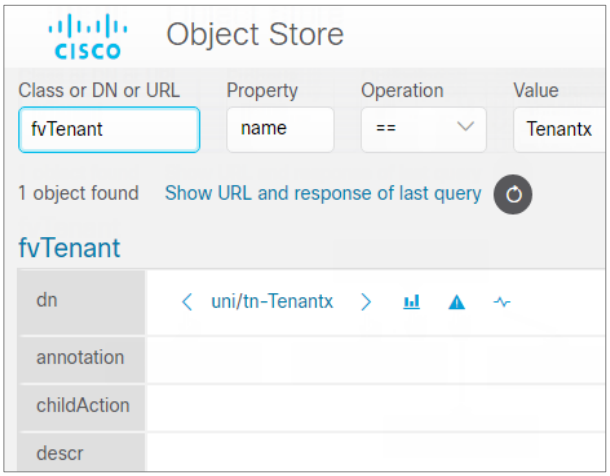
Visore

(DCACIO v4.0)

Visore – APIC Object Store Browser

Visore provides a graphical view of managed objects (MOs) using a browser





REST stands for **Representational State Transfer**. It relies on a stateless client to server communication protocol that mostly uses HTTP as it's transport. API's that adhere to the core principles of REST do not require the client to know much of structure of the API it is interacting with.

The ACI programmatic interface is a REST interface and behind that interface is the MIT that we have talked about. To interact with that object store we can use various tools. You can find many software applications available for all platforms to interact with REST interfaces. Two particular tools are available for ACI users:

- **Visore** - An object browser built into the ACI Application Policy Infrastructure Controller (APIC). Visore is Italian for Viewer.
- **Postman** - A very popular REST interface browser available on many platforms including Chrome

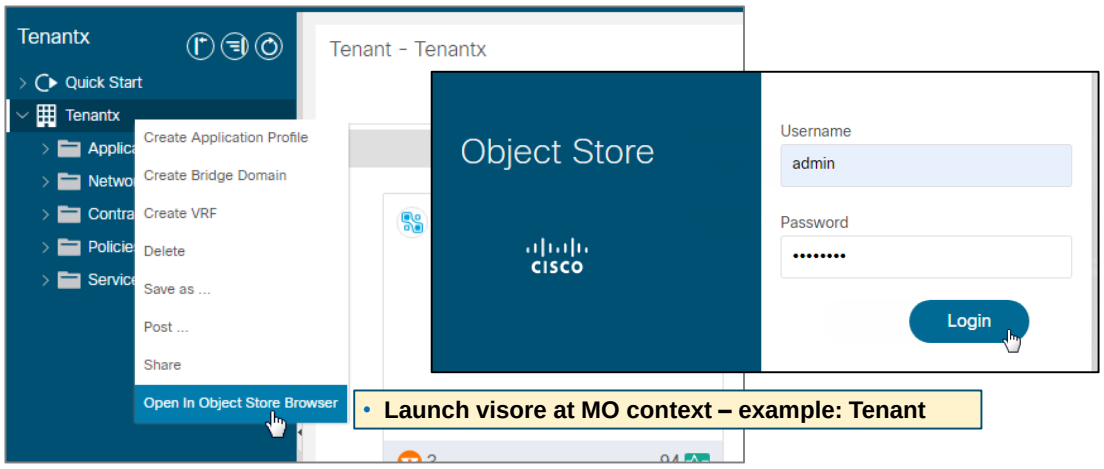
As part of the APIC controller tools, Visore is a front end to the object model that is running on the ACI fabric. Visore is the best tool for browsing the Cisco ACI object store on an APIC. Unlike Postman, Visore cannot do is update or make changes to the objects in the ACI fabric. Postman can be used to both query and configure APIC.

Visore provides a graphical view of the managed objects (MOs) using a browser. The Visore utility uses the APIC REST API query methods to browse MOs active in the Application Centric Infrastructure fabric, allowing you to see the query that was used to obtain the information. Visore provides you with a easy link to see the children of the query class and also get a link directly to the API documentation.

Using Visore provides an easy method to browse the parent/child relationships of objects in the management information tree. In the DN element of the object there are two arrows to the left and right. These reference the parent and the child element of the object.

Accessing Visore

You can launch visore from browser URL: `https://10.1.1.1/visore.html`



Visore requires its own separate login to APIC. There are multiple methods to launching Visore. The diagram illustrates selecting a managed object (MO), in this case a tenant, to Open the Object Store Browser (aka Visore). Once open, your are placed at the selected object's location in the Management Information Tree (MIT).

Another method to open Visore from APIC is to select the gearwheel icon (in the GUI upper right-hand corner) and choose **Object Store Browser** (not illustrated).

- This will launch Visore from the root of the MIT

You can also launch Visore by entering the following in a supported browser URL:

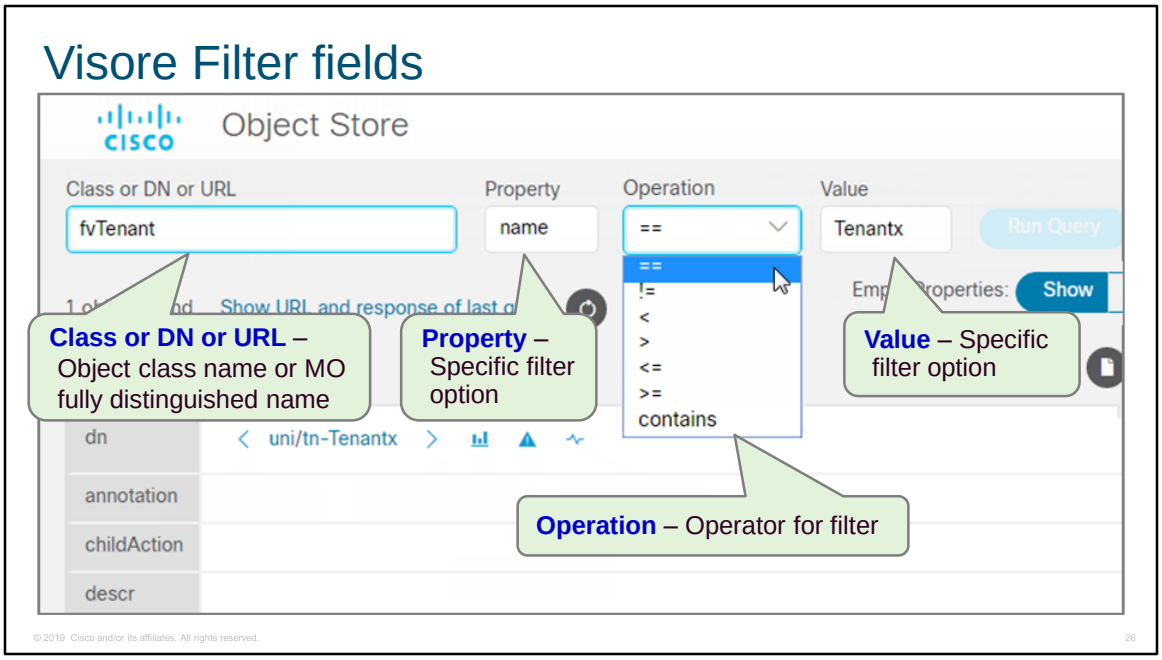
- https://<APIC IP>/visore.html
- When prompted, log in using the same credentials you would use to log in to the APIC CLI or GUI user interfaces.
- You can use a read-only account.

Note: Only the Firefox, Chrome, and Safari browsers are supported for Visore access.

Once enabled, when you click around the GUI you will notice the text shown in this

image in the bottom left hand corner. You can then use this in order to find the necessary information. You are interested in the latter half of the text, which begins with 'Current MO'. This is the current MO that is displayed.

https://www.cisco.com/c/en/us/td/docs/switches/datacenter/aci/apic/sw/1-x/api/rest/b_APIC_RESTful_API_User_Guide/b_IFC_RESTful_API_User_Guide_chapter_0100.html#concept_A322FE0064B046D1A8D9AAD5D398A23B



Visore Filters

Previously when you searched on the Class type of 'fvAEPg' it returned a result of 1615 objects, but you can narrow this search.

For example, perhaps you are only interested in the objects that are within the Application Profile 'ap-default'. As this is contained within the DN.

Class or DN or URL:

In order to execute a query, enter either the Class or DN or MO URL.

Property Field:

The property of the managed object on which you want to filter the results. If you leave the **Property**field empty, the search returns all instances of the specific class.

Discovery Lab 7 – Visore

(DCACIO v4.0)

CLI Troubleshooting Commands

(DCACIO v4.0)

Managed Object Query – moquery

```
apic1# moquery -c
usage: Command line cousin to visore [-h] [-i HOST] [-p PORT] [-d DN]
----      [-c KLASS] [-f FILTER] [-a ATTRS]
----      [-o OUTPUT] [-u USER]
----      [-x [OPTIONS [OPTIONS ...]]]
Command line cousin to visore: error: argument -c/--klass: expected one arg

apic1# moquery -c fvAp --dn uni/tn-Tenant34
Total Objects shown: 1
# fv.Ap
name- : POC
childAction  :
descr-:
dn-    : uni/tn-Tenant34/ap-POC
lcOwn-: local
modTs-: 2016-03-19T08:28:50.217+00:00
monPolDn      : uni/tn-common/monepg-default
  ... Output truncated
```

-c = Specifies a class name for the query

Managed Object Query – moquery (the command line cousin to Visore) allows administrators to query a specific object by its distinguished name (dn), or report all objects of a particular object class.
For example, report all tenants by querying the fvTenant class:
apic1# moquery -c fvTenant

Moquery Help:
apic1# **man moquery**

NAME:
moquery -- search for MO

SYNOPSIS:
moquery --help --host hostname --port portname --dn dn --klass class-name --filter property --attrs attributes --output output --user user-name --options options

DESCRIPTION: Searches for managed objects (MOs) within the management information tree (MIT).

- -h, --help: Displays usage information.
- -i, --host: Specifies an APIC host.
- hostname: The hostname or IP address of an APIC.
- -p, --port: Specifies a port for a REST interface.
- Portname: The REST interface port number.
- -d, --dn: The DN of an MO.
- -c, --klass: Specifies a class name for the query.
- classname: Specifies a class. You can enter multiple classes separated by commas.
- -f, --filter: Specifies a property on which to filter MOs.
- -property: The property on which to filter MOs.
- -a, --attrs: Specifies the attributes that the query displays.

Attributes - The type of attributes to display. You can choose config or all. If config is selected, only configurable attributes are displayed. Unless the 'table' output format is specified, the default is all.

- -o, --output: Specifies the output format of the query results.
- -u, --user: Specifies a user name.
- Username: the user name.
- -x, --options: Specifies query options.

Options - The query options to enable. You can specify options as supported by the REST API. You can add multiple options statements to the command, using syntax such as the following:

-x [OPTIONS [OPTIONS ...]] [-x [OPTIONS [OPTIONS ...]]].

For example:

moquery -c firmwareCtrlrFwStatusCont -x query-target=subtree target-subtree-class=firmwareCtrlrRunning

Viewing Faults Using moquery

Moquery can be used to report faults in **text**, **xml**, or **json** formats for later analysis

- **Save report to ascii file on leaf filesystem:**

```
leaf# moquery -c faultInst > /tmp/fault-20141112.txt
leaf# ls -l /tmp/fault-20141112.txt
-rw-------1 admin admin 40113 Nov 13 13:37 /tmp/fault-20141112.txt
```

- **Report the cause and distinguished name for a given fault:**

```
leaf# moquery -c faultInst -f 'fault.Inst.code == "F0467"' | egrep "cause|dn"
cause : configuration-failed
dn: uni/epp/fv-[uni/tn-testTenant2/ap-testAP/epg-testEPG]/nwissues/fault-F0467
```


Using the more traditional NX-OS style commands in the APIC CLI can present specific information in a text format. In addition, administrators can query the APIC object database directly using the CLI command "**moquery**".

Viewing Faults Using moquery
Moquery tool can be used to report faults in text, xml, or json format for later analysis. This command is referred to as the command-line cousin to Visore. Moquery requires you first SSH to the APIC. More on moquery command options are presented later in this lesson.

Examples:

```
moquery -c faultInst -f 'fault.Inst.code=="F0467"'
moquery -c faultRecord -x order-
by="faultRecord.created|desc" 'query-target-
filter=wcard(faultRecord.created,"2017-12-
1[2]")'>/home/admin/auditlog.txt
```

Output moquery command to json:

```
leaf1# moquery -c faultInst -o json
```

```
        {
         "imdata": [
          {
           "faultInst": {
                          "attributes": {
                                          "dn": "sys/phys-
[eth1/11]/fault-F1186",
                                          "domain":
"infra",
                                          "code": "F1186",
                                          "occur": "1",
                                          "subject":
"failure-to-deploy",
                                          "severity":
"warning",
                                          "descr": "Port
configuration failure.
```

moquery Object Specification

- Moquery syntax to query specific managed objects

Sample command to report all data for a given fault:

```
apic1# moquery -c faultInst -f 'fault.Inst.code=="F0721"'
Total Objects shown: 1

# fault.Inst
code              : F0721
ack               : no
cause             : configuration-failure
changeSet         : allocMode:static, configIssues:missing-encapblk, name:Tx-L3ext-Vlan
childAction       :
created           : 2018-04-08T16:05:54.592+00:00
delegated         : no
descr             : VLAN/VxLAN/NVGRE pool Tx-L3ext-Vlan deployment failed due to: Invalid
 or missing Encapsulation Blocks.
dn                : uni/infra/vlanns-[Tx-L3ext-Vlan]-static/fault-F0721
domain            : infra
highestSeverity   : minor
lastTransition    : 2018-04-08T16:08:10.194+00:00
lc                : raised
modTs             : never
occur             : 1
origSeverity      : minor
prevSeverity      : minor
rn                : fault-F0721
rule              : fvns-ainst-pconfig-failed
severity          : minor
status            :
subject           : infra-policy
type              : config
uid               :
```

```
BRIDGE DOMAIN
moquery -c fvBD
moquery -c fvBD -f "fv.BD.name==\"BDname\""

CONTEXT
moquery -c fvCtx

EPG
moquery -c fvAEPg
moquery -c fvAEPg -f 'fv.AEPg.pcTag=="xxxx"'

ENDPOINT
moquery -c fvCEp
moquery -c fvCEp | grep x.x.x.x -A 10 -B 5

CONSUMED CONTRACT
moquery -c vzBrCP

PROVIDED CONTRACT
moquery -c vzBrCP

L3 OUT
moquery -c l3extInstP
moquery -c l3extDomP

FAULT
moquery -c faultInst
```


moquery Object Specification

Examples of common moquery syntax to query specific managed objects, including a fault instance.

moquery – EPG and Bridge Domain examples

- View learned end-points using a given subnet

```
apic1# moquery -c fvCEp | grep 192.168.10.1  -A 3
ip           : 192.168.10.11
lcC          : learned,vmm
lcOwn        : local
mac          : 00:50:56:AC:31:25
--
ip           : 192.168.10.12
lcC          : learned,vmm
lcOwn        : local
mac          : 00:50:56:A8:C7:B0
apic1#
apic1# moquery -c fvCEp | grep 192.168.11.1  -A 3
ip           : 192.168.11.11
lcC          : learned,vmm
lcOwn        : local
mac          : 00:50:56:AC:C2:0D
--
ip           : 192.168.11.12
lcC          : learned,vmm
lcOwn        : local
mac          : 00:50:56:A8:9F:DE
apic1#
apic1# moquery -c fvCEp | grep 192.168.12.1  -A 3
ip           : 192.168.12.11
lcC          : learned,vmm
lcOwn        : local
mac          : 00:50:56:AC:0B:7E
apic1#
```

- View Bridge Domain data

```
apic1# moquery -d uni/tn-Tenantx/BD-App
# fv.BD
name                          : App
OptimizeWanBandwidth          : no
annotation                    :
arpFlood                      : yes
bcastP                        : 225.1.109.80
childAction                   :
configIssues                  :
descr                         :
dn                            : uni/tn-Tenantx/BD-App
epClear                       : no
epMoveDetectMode              :
extMngdBy                     :
hostBasedRouting              : no
intersiteBumTrafficAllow      : no
intersiteL2Stretch            : no
ipLearning                    : yes
lcOwn                         : local
limitIpLearnToSubnets         : yes
llAddr                        : ::
mac                           : 00:22:BD:F8:19:FF
mcastAllow                    : no
modTs                         : 2019-04-17T09:17:09.555+00:00
monPolDn                      : uni/tn-common/monepg-default
mtu                           : inherit
multiDstPktAct                : bd-flood
```


moquery – EPG and Bridge Domain examples

- View learned end-points using a given subnet

  APIC# **moquery -c fvCEp | grep 192.168.10.1 -A 3**

- View a given Bridge Domain's settings

  APIC# **moquery -d uni/tn-Tenantx/BD-App**

Discover APIC Initial Setup Values Using moquery

moquery can be used to report APIC initial setup values

```
Cluster configuration ...
  Enter the fabric name [ACI Fabric1]:

  Enter the controller name [apic1]:
  Enter address pool for TEP addresses [10.0.0.0/16]:

  Enter the VLAN ID for infra network (1-4094) [4]:
  Enter address pool for BD multicast addresses (GIPO) [225.0.0.0/15]:

Out-of-band management configuration ...
  Enable IPv6 for Out of Band Mgmt Interface? [N]:
  Enter the IPv4 address [192.168.10.1/24]: 192.168.10.100/24
  Enter the IPv4 address of the default gateway [192.168.10.254]: 192.168.10.1

admin user configuration ...
  Enable strong passwords? [Y]: N
  Enter the password for admin: _
```


Key Parameter Values used during APIC initial setup:

- Fabric name
- Number of controllers
- Controller ID
- IP address pool for tunnel endpoint addresses (TEP)
- IP address pool for bridge domain multicast address (GIPO)
- Management interface speed/duplex mode
- VLAN ID for infrastructure network
- IPv4/IPv6 addresses for the out-of-band management
- IPv4/IPv6 addresses of the default gateway
- Strong password check

Discover APIC Initial Setup Values Using moquery

Find the **Fabric Domain Name** configured on each APIC in Cluster:

```
apic# moquery -c infraCont | grep -E "dn|fbDmNm|size"
```

```
apic1# moquery -c infraCont | grep -E "dn|fbDmNm|size"
dn           : topology/pod-1/node-1/av
fbDmNm       : ACI Fabric1
size         : 1
```

Find the **TEP Pool** and **Pod ID** for the Fabric:

```
apic# moquery -c fabricSetupP | grep -E "podId|tepPool"
```

```
apic1# moquery -c fabricSetupP | grep -E "podId|tepPool"
podId        : 1
tepPool      : 10.0.0.0/16
apic1#
```


Moquery commands to discover the configuration settings used during APIC1 initial setup script.

Find the Fabric Domain Name configured on each APIC in Cluster:

```
apic# moquery -c infraCont | grep -E "dn|fbDmNm|size"
```

Find the TEP Pool and Pod ID for the Fabric:

```
apic# moquery -c fabricSetupP | grep -E "podId|tepPool"
```

Discover APIC Initial Setup Values Using moquery

Find the **Multicast Address** range pool

```
apic# moquery -c fvBD | grep -E "name|bcastP|dn" | grep -B 2 "Tenantx"
```

```
apic1# moquery -c fvBD | grep -E "name|bcastP|dn" | grep -B 2 "Tenantx"
name                      : App
bcastP                    : 225.1.109.80
dn                        : uni/tn-Tenantx/BD-App
--
name                      : Web
bcastP                    : 225.1.15.64
dn                        : uni/tn-Tenantx/BD-Web
```

Strong password check configuration settings

```
apic# moquery -c aaaUserEp | grep "pwdStrengthCheck"
```

```
apic1# moquery -c aaaUserEp | grep "pwdStrengthCheck"
pwdStrengthCheck : no
```


Moquery commands to discover the configuration settings used during APIC1 initial setup script.

Find the Multicast Address range pool:

```
apic# moquery -c fvBD | grep -E "name|bcastP|dn" | grep -B 2 "Tenantx"
```

Strong password check configuration settings:

```
apic# moquery -c aaaUserEp | grep "pwdStrengthCheck"
```

Using moquery on Leaf Switches

Moquery can report specific switch information

- Report the VLAN associated with an interface:

```
leaf101# moquery -c nwPathEp -f 'nw.PathEp.id=="eth1/8"' | grep Encap
```

```
apic1# moquery -c nwPathEp -f 'nw.PathEp.id=="eth1/8"' | grep Encap
nativeEncap              : vlan-3498
```

- Report which EPGs are configured to use a port on a given leaf:

```
apic# moquery -c fvIfConn | grep dn | grep eth1/16"
```

```
apic1# moquery -c fvIfConn | grep dn | grep eth1/16
dn : uni/epp/fv-[uni/tn-Tenantx/ap-POC/epg-DB]/node-102/stpathatt-[eth1/16]/
conndef/conn-[vlan-1399]-[0.0.0.0]
```


What other EPG are configured to use Leaf 103 port 1/8

```
Leaf103# moquery -c fvIfConn | grep dn | grep eth1/8
dn : resPolCont/rtdOutCont/rtdOutDef-[uni/tn-Tenant8/out-
L3ext-L103]/node-103/stpathatt-[eth1/8]/conndef/conn-
[vlan-2498]-[10.99.99.2/24]
```

What VLAN is associated with an interface?

```
leaf101# moquery -c nwPathEp -f 'nw.PathEp.id=="eth1/16"'
Total Objects shown: 1

# nw.PathEp
id : eth1/16
allocState : allocated
childAction :
descr :
dn : sys/conng/path-[eth1/16]
fabricPathDn : topology/pod-1/paths-101/pathep-[eth1/16]
lcOwn : local
modTs : 2018-03-26T05:51:53.874+00:00
```

```
monPolDn : uni/fabric/monfab-default
name :
nativeEncap : vlan-1940
rn : path-[eth1/16]
status :
vlanScope : global
vlanScopeSupport : supported
```

Discovery Lab 8 – moquery

(DCACIO v4.0)

APIC Controller Status – acidiag fnvread

Built-in acidiag command to check status of controllers and fabric nodes

```
apic1# acidiag fnvread
ID     Name            Serial Number   IP Address       Role    Pod ID  State    LastUpdMsgId
-------------------------------------------------------------------------------------------
101    pod2-leaf1      SAL1820SMHV     10.0.168.95/32   leaf    1       active   0
102    pod2-leaf2      SAL1816QVBC     10.0.168.93/32   leaf    1       active   0
103    pod2-leaf3      SAL1818RUHM     10.0.168.91/32   leaf    1       active   0
104    pod2-leaf4-BIS  SAL1818RP59     10.0.56.95/32    leaf    1       active   0
201    pod2-spine1     SAL1811NN5S     10.0.168.92/32   spine   1       active   0
202    pod2-spine2     SAL1811NN5X     10.0.168.94/32   spine   1       active   0
```

- acidiag can be used to verify all processes are running on the switch
- The **acidiag fnvread** report displays similar output as the APIC GUI:
  - **Fabric** | **Inventory** | **Fabric Membership**

Performing APIC OS commands – **acidiag [fnvread** | **avread** | **rvread**]

- Fabric node vector (fnvread): Vector of switch nodes including address and state (see fabric discovery earlier section)
- Appliance vector (avread): Vector of APICs with their address, state
- Replica vector (rvread): Vector of shards and replicas including leader for each shard

This state information is distributed and updated by appliance director **Data Management Engine** (**DME**) to each node. The DME is a service that runs on the APIC that manages data for the data model. DMEs communicate using message entities called "stimulus", which usually have a request and response. Each service on the APIC or switch is built over a library or framework called DME.

The APIC built-in acidiag command can be employed check status of controllers and fabric nodes.
The **acidiag fnvread** report displays similar output as the APIC GUI | Fabric | Inventory | Fabric Membership

APIC Cluster Verification – acidiag avread

```
apic1# acidiag aread
Local appliance ID=1 ADDRESS=10.0.0.1 TEP ADDRESS=10.0.0.0/16 CHASSIS_ID=e52e3fc8-4cfa-11e6-af0d-81a89d6fa5ec
Cluster of 3 lm(t):1(2018-09-21T07:31:09.458+00:00) appliances (out of targeted 3 lm(t):1(2018-09-
21T07:31:12.350+00:00)) with FABRIC_DOMAIN name=POD13 set to version=apic-4.0(1h) lm(t):1(2018-09-
21T07:31:22.763+00:00); discoveryMode=PERMISSIVE lm(t):0(1970-01-01T00:00:00.003+00:00)
  appliance id=1  address=10.0.0.1 lm(t):1(2018-09-21T07:30:45.112+00:00) tep address=10.0.0.0/16 lm(t):1(2018-09-
21T07:30:45.112+00:00) oob address=10.48.22.94/24 lm(t):1(2018-09-21T07:31:09.700+00:00) version=4.0(1h)
lm(t):1(2018-09-21T07:31:10.690+00:00) chassisId=e52e3fc8-4cfa-11e6-af0d-81a89d6fa5ec lm(t):1(2018-09-
21T07:31:10.690+00:00) capabilities=0X7FFFFFFF--0--0X7 lm(t):1(2018-09-21T07:36:22.835+00:00)
rK=(stable,present,0X207373642D687373) lm(t):1(2018-09-21T07:31:09.707+00:00) aK=(stable,present,0X207373642D687373)
lm(t):1(2018-09-21T07:31:09.707+00:00) cntrlSbst=(APPROVED, FCH1852V3LE) lm(t):1(zeroTime) commissioned=YES
lm(t):1(zeroTime) registered=YES lm(t):1(2018-09-21T07:30:45.112+00:00) active=YES(2018-09-21T07:30:45.112+00:00)
health=(applnc:255 lm(t):1(2018-09-21T07:32:54.051+00:00) svc's)
  appliance id=2  address=10.0.0.2 lm(t):1(2018-09-21T07:31:21.488+00:00) tep address=10.0.0.0/16 lm(t):2(2018-09-
18T14:49:06.317+00:00) oob address=10.48.22.95/24 lm(t):1(2018-09-01T11:31:23.647+00:00) version=4.0(1h)
lm(t):2(2018-09-21T07:31:10.698+00:00) chassisId=4b00064a-3146-11e6-bf20-ed9eb8a4f642 lm(t):1(2018-09-
21T07:31:21.488+00:00) capabilities=0X7FFFFFFF--0--0X7 lm(t):2(2018-09-21T07:34:44.120+00:00)
rK=(stable,present,0X207373642D687373) lm(t):1(2018-09-21T07:31:11.958+00:00) aK=(stable,present,0X207373642D687373)
lm(t):1(2018-09-21T07:31:11.958+00:00) cntrlSbst=(APPROVED, FCH1906V223) lm(t):0(zeroTime) commissioned=YES
lm(t):1(2018-09-21T07:31:09.458+00:00) registered=YES lm(t):1(2018-09-18T15:19:22.856+00:00) active=YES(2018-09-
21T07:31:10.534+00:00) health=(applnc:255 lm(t):2(2018-09-21T07:32:53.974+00:00) svc's)
  appliance id=3  address=10.0.0.3 lm(t):1(2018-09-21T07:31:21.488+00:00) tep address=10.0.0.0/16 lm(t):3(2018-06-
13T16:49:15.233+00:00) oob address=10.48.22.96/24 lm(t):1(2018-09-01T11:31:23.654+00:00) version=4.0(1h)
lm(t):3(2018-09-21T07:31:10.668+00:00) chassisId=aa08b322-3186-11e6-87c2-7be412f9e01b lm(t):1(2018-09-
21T07:31:21.488+00:00) capabilities=0X7FFFFFFF--0--0X7 lm(t):3(2018-09-21T07:35:09.293+00:00)
rK=(stable,present,0X207373642D687373) lm(t):1(2018-09-21T07:31:11.960+00:00) aK=(stable,present,0X207373642D687373)
lm(t):1(2018-09-21T07:31:11.961+00:00) cntrlSbst=(APPROVED, FCH1904V2RP) lm(t):1(2018-06-13T16:49:40.020+00:00)
commissioned=YES lm(t):1(2018-09-21T07:31:09.458+00:00) registered=YES lm(t):1(2018-09-18T15:19:22.856+00:00)
active=YES(2018-09-21T07:31:10.421+00:00) health=(applnc:255 lm(t):3(2018-09-21T07:32:54.182+00:00) svc's) ..
```

Collect **acidiag avread** from all APICs – invoke on all APIC controllers.
What cluster issues to compare and check:

- Fabric name must be identical
- Chassis Id for each appliance should identical on the 3 apics
- Cluster version must be identical on each appliance and must match cluster version
- Health **255** (means replica are all fine)

APIC Certificate Verification – acidiag verifyapic

Built-in acidiag command which will check certificate validity on APIC

Factors that can cause certs to be invalid include:

- NTP/time as certs are typically valid within a 10 year window
- TPM Programming (done in mfg)
- Internal USB which has keyfile (do not remove)
- TPM ownership (do not reset BIOS settings)
- PID/SN check (cannot turn any C220 into an APIC)

Good!

```
apic1# acidiag verifyapic
openssl_check: certificate details
subject= CN=FCH2048V28U,serialNumber=PID:APIC-SERVER-M2 SN:FCH2048V28U
issuer= CN=Cisco Manufacturing CA,O=Cisco Systems
notBefore=Dec 19 05:20:45 2016 GMT
notAfter=Dec 19 05:30:45 2026 GMT
openssl_check: passed
ssh_check: passed
all_checks: passed
```

Bad!

```
Standalone-APIC# acidiag verifyapic
file not found: /securedata/ssl/server.crt
installation_check.sh ERROR: openssl_check: files check failed
```


The APIC built-in **acidiag** command can be employed to check certificate validity on the APIC.

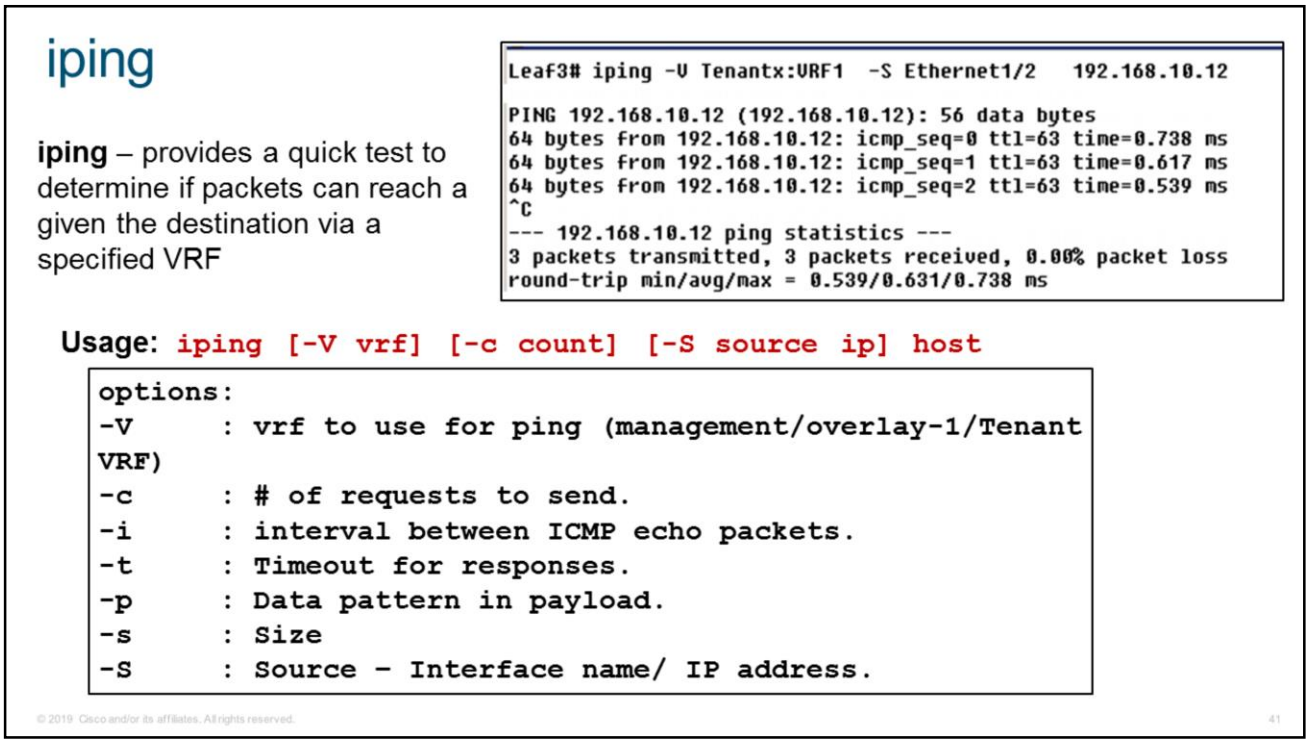
If state is inactive but with an IP allocated likely Certificate issue :

- HW clock Mismatch
- Certificate bad or missing

Factors that can cause certs to be invalid include:

- NTP/time as certs are typically valid within a 10 year window
- TPM Programming (done in mfg)
- Internal USB which has keyfile (do not remove)
- TPM ownership (do not reset BIOS settings)
- PID/SN check (cannot turn any C220 into an APIC)

This diagram illustrates both a good command result and a failed check.



iping - Fabric node command used in place of ping/traceroute,

- Employ to send ping out data plane
- Standard ping sent out OOB

The iping command provides a quick test to determine if packets can reach a particular the destination via the specified VRF.

```
leaf# iping -h
usage: iping [-dDFLnqRrv] [-V vrf] [-c count] [-i wait] [-p pattern] [-s
packetsize] [-t timeout] [-S source ip/interface] host
```

Example: source a ping from a given VRF on a Leaf

```
leaf# iping 10.255.118.27 -V common:Common-VRF
```

Additional examples: ping an IP across the infra-Vlan

```
pod1-leaf1# iping -V overlay-1 10.0.59.154

PING 10.0.59.154 (10.0.59.154): 56 data bytes
64 bytes from 10.0.59.154: icmp_seq=0 ttl=55 time=0.254 ms
64 bytes from 10.0.59.154: icmp_seq=1 ttl=55 time=0.256 ms
64 bytes from 10.0.59.154: icmp_seq=2 ttl=55 time=0.245 ms
64 bytes from 10.0.59.154: icmp_seq=3 ttl=55 time=0.241 ms
64 bytes from 10.0.59.154: icmp_seq=4 ttl=55 time=0.23 ms

--- 10.0.59.154 ping statistics ---
```

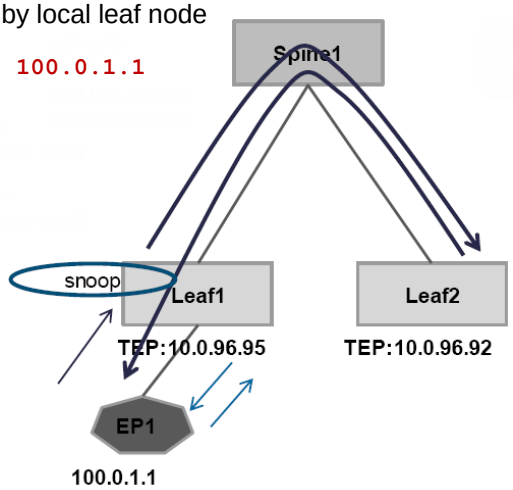
```
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min/avg/max = 0.23/0.245/0.256 ms
```

iping Internal

- Recommend set source IP address to make clear which address is used
- ICMP echo reply packet to remote leaf is relayed by local leaf node

```
Leaf2# iping -V Prod:VRF1 -S 100.0.1.254  100.0.1.1
```

Tenant: Prod
VRF: VRF1
Subnet: 100.0.1.254/24
--100.0.1.254/24
→
iping from leaf1
→
iping from leaf2



iping Internal
- Recommend set source IP address to make clear which address is used
- ICMP echo reply packet to remote leaf is relayed by local leaf node

itraceroute

itraceroute – provides the following improvements over traditional traceroute

- Discovers and reports multiple paths
- Transits only a single probe packet per path
- Reports detailed node information
- Simulates tenant traffic, exploring paths under the applied policies

```
Leaf103# itraceroute
 A.B.C.D    Enter destination IP
 A:B::C:D   Enter destination IPv6
 external   Run itraceroute with 5-Tuple
 src-ip     Source IPv6

Leaf103# itraceroute 192.168.12.11
 <CR>       Carriage return
 payload    payload size
 vrf        tenant vrf
```


Itraceroute features:

- Discovers and reports multiple paths
- Transits only a single probe packet per path
- Reports detailed node information
- Simulates tenant traffic, exploring paths under the applied policies

SYNTAX: itraceroute target_ip vrf source_tenant:ctx

Eaxmple:

```
leaf4# itraceroute 62.1.1.40 vrf common:deadbeef-net1
Tenant traceroute to 62.1.1.40, tenant VRF common:deadbeef-net1, source encap
vlan-61, from [65.1.1.1], payload 56 bytes

Path 1    [ Complete ]    [ internal ]
   1: TEP      10.0.192.93  intf    eth1/4  1.014 ms
   2: TEP      10.0.192.90  intf unspecified  3.471 ms

Path 2    [ Complete ]    [ internal ]
   1: TEP      10.0.192.94  intf    eth1/4  0.942 ms
   2: TEP      10.0.192.90  intf unspecified  3.577 ms
```

itraceroute

Usage: `itraceroute destination-ip vrf tenant-vrf`

```
Leaf103# itraceroute 192.168.12.11 vrf Tenant8:VRF1
Tenant traceroute to 192.168.12.11, tenant VRF Tenant8:VRF1, source encap vlan-1380, from [192.168.12.1],
payload 56 bytes
. . .
Path 1   [ Complete ]   [ internal ]
+-----+-------------+---------+------+-------------+------+
| Hop |     TEP     |   ETEP  | Site |  Interface  | Time |
+-----+-------------+---------+------+-------------+------+
|  1  | 10.0.112.65 | 0.0.0.0 |  0   |    eth1/1   | 0.0  |
|  2  | 10.0.112.64 | 0.0.0.0 |  0   | unspecified | 0.0  |
+-----+-------------+---------+------+-------------+------+

Path 2   [ Complete ]   [ internal ]
+-----+-------------+---------+------+-------------+-------+
| Hop |     TEP     |   ETEP  | Site |  Interface  |  Time |
+-----+-------------+---------+------+-------------+-------+
|  1  | 10.0.112.65 | 0.0.0.0 |  0   |    eth1/1   | 0.001 |
|  2  | 10.0.112.67 | 0.0.0.0 |  0   | unspecified |  0.0  |
+-----+-------------+---------+------+-------------+-------+

Leaf103#
```


Itraceroute example

Node iTraceroute – Example:

```
pod2-leaf1# itraceroute 10.0.40.95

Node traceroute to 10.0.40.95, infra VRF overlay-1, from [10.0.40.66],
payload 56 bytes
Path 1
1: TEP 10.0.64.64 intf eth1/33 0.611 ms
2: TEP 10.0.40.95 intf eth1/98 0.608 ms
Path 2
1: TEP 10.0.64.65 intf eth1/33 0.473 ms
2: TEP 10.0.40.95 intf eth1/97 0.540 ms
```

Discovery Lab 9 – iping / itraceroue

(DCACIO v4.0)

Troubleshooting Endpoint Connectivity

(DCACIO v4.0)

Troubleshooting Endpoint Connectivity

- **What are the current Endpoints in the fabric?**
- **Where is a specific Endpoint?**
- **What was connected to the network last Thursday between 3:30 and 4:00?**
- **What are all of the Endpoints belonging to a given Tenant?**
- **What Endpoints are on this subnet?**
- **What is the history of a given Endpoint (i.e. movement, etc.)?**

Troubleshooting Endpoint Connectivity

Typical queries around endpoint connectivity:

- What are the current Endpoints in the fabric?
- Where is a specific Endpoint?
- What was connected to the network last Thursday between 3:30 and 4:00?
- What are all of the Endpoints belonging to a given Tenant?
- What Endpoints are on this subnet?
- What is the history of a given Endpoint (i.e. movement, etc.)?

.

Endpoints in Cisco ACI

Traditional networks maintain external device addresses using three tables

Cisco ACI maintains this information in a different way:

- Endpoints consist of one MAC address and zero or more IP addresses
- Each leaf is responsible for reporting its local endpoints to the **Council Of Oracle Protocol** (**COOP**) database, located on each spine switch
- Local endpoints are the main source of endpoint information for entire ACI fabric

Traditional network	
Table	**Table role**
RIB	• IPv4 addresses (/32 and non-/32) • IPv6 addresses (/128 and non-/128)
MAC address table	MAC addresses
ARP table	Relationship of IP to MAC

Cisco ACI	
Table	**Table role**
RIB	• IPv4 (non-/32*) • IPv6 (non-/128*)
Endpoint	MAC and IP addresses (/32 or /128 only)
ARP	Relationship of IP to MAC (only for Layer 3 outside [L3Out] connections)

High Level on Endpoint Learning

In traditional Network:

- Mac-address table updated by dataplane
- ARP table updated by ARP/GARP
- IPv6 ND table updated by ND messages

In ACI:

- No separate mac-address table and arp table
- Host identity stored as endpoint
- Endpoint can be layer 2 or layer 2 + layer 3 information
- Endpoint entries updated by dataplane and ARP/GARP/ND by default
- Local learns point to attached EP, remote learns point to tunnel

Cisco ACI and endpoints

In a traditional network, three tables are used to maintain the network addresses of external devices:

- MAC address table for Layer 2 forwarding
- Routing Information Base (RIB) for Layer 3 forwarding
- ARP table for the combination of IP addresses and MAC addresses.

Cisco ACI maintains this information in a different way.

- Cisco ACI uses endpoints to forward traffic
- An endpoint consists of one MAC address and zero or more IP addresses
- Each endpoint represents a single networking device.

As the Tables show, Cisco ACI replaced the MAC address table and ARP table with a single table called the endpoint table. This change implies that Cisco ACI learns that information in a different way than in a traditional network. Cisco ACI learns MAC and IP addresses in hardware by looking at the packet source MAC address and source IP address in the data plane instead of relying on ARP to obtain a next-hop MAC address for IP addresses. This approach reduces the amount of resources needed to process and generate ARP traffic. It also allows detection of IP address and MAC address movement without the need to wait for GARP as long as some traffic is sent from the new host.

Local endpoints and remote endpoints

A leaf switch has two types of endpoints: local endpoints and remote endpoints. Local endpoints for LEAF1 reside directly on LEAF1 (For example, directly attached), whereas remote endpoints for LEAF1 reside on other leaf endpoints.

Although both local and remote endpoints are learned from the data plane, remote endpoints are merely cached locally to each leaf. Local endpoints are the main source of endpoint information for the entire Cisco ACI fabric. Each leaf is responsible for reporting its local endpoints to the Council Of Oracle Protocol (COOP) database, located on each spine switch, which implies that all endpoint information in the Cisco ACI fabric is stored in the spine COOP database. Because this database is accessible, each leaf does not need to know about all the remote endpoints to forward packets to the remote leaf endpoints. Instead, a leaf can forward packets to spine switches, even if the leaf does not know about a particular remote endpoint. This forwarding behavior is called spine proxy.

Local endpoint learning

Cisco ACI learns the MAC (and IP) address as a local endpoint when a packet comes into a Cisco ACI leaf switch from its front-panel ports. Front-panel ports are southbound ports from the perspective of Cisco ACI and do not face spine switches.

A Cisco ACI leaf switch follows these steps to learn a local endpoint MAC address and IP address:

1. Leaf receives a packet with a source MAC Address (MAC A) and source IP Address (IP A).
2. Leaf learns MAC A as a local endpoint.
3. Leaf learns IP A tied to MAC A if the packet is an ARP packet.
4. Leaf learns IP A tied to MAC A if the packet is routed.

If the packet is switched and not an ARP packet, the Cisco ACI leaf never learns the IP address but only the MAC address. This behavior is the same as traditional MAC address learning behavior on a traditional switch

Remote endpoint learning

Cisco ACI learns a MAC or IP address as a remote endpoint when a packet comes into a Cisco ACI leaf switch from another leaf switch through a spine switch. When a packet is sent from one leaf to another leaf, Cisco ACI encapsulates the original packet with an outer header representing the source and destination leaf Tunnel Endpoint (TEP) and the Virtual Extensible LAN (VXLAN) header, which contains the bridge domain or VRF information of the original packet.

Packets that are switched contain bridge domain information. Packets that are routed contain VRF information.

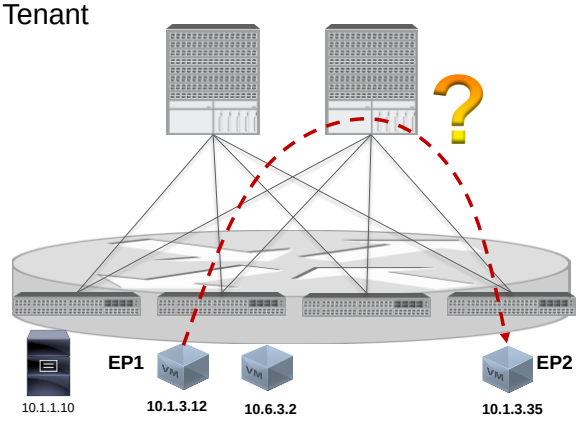
A Cisco ACI leaf switch follows these steps to learn a remote endpoint MAC or IP address:

1. Leaf receives a packet with source MAC A and source IP A from a spine switch.
2. Leaf learns MAC A as a remote endpoint if VXLAN contains bridge domain information.
3. Leaf learns IP A as a remote endpoint if VXLAN contains VRF information.

A given packet is Layer 3 traffic with the Cisco ACI bridge domain Switch Virtual Interface (SVI) as its default gateway. Therefore, both the MAC address and IP address (Src MAC S and Src IP) are learned as a single local endpoint on the source leaf, and only IP addresses are learned as a remote endpoint on the destination leaf.

Endpoint Troubleshooting Tasks

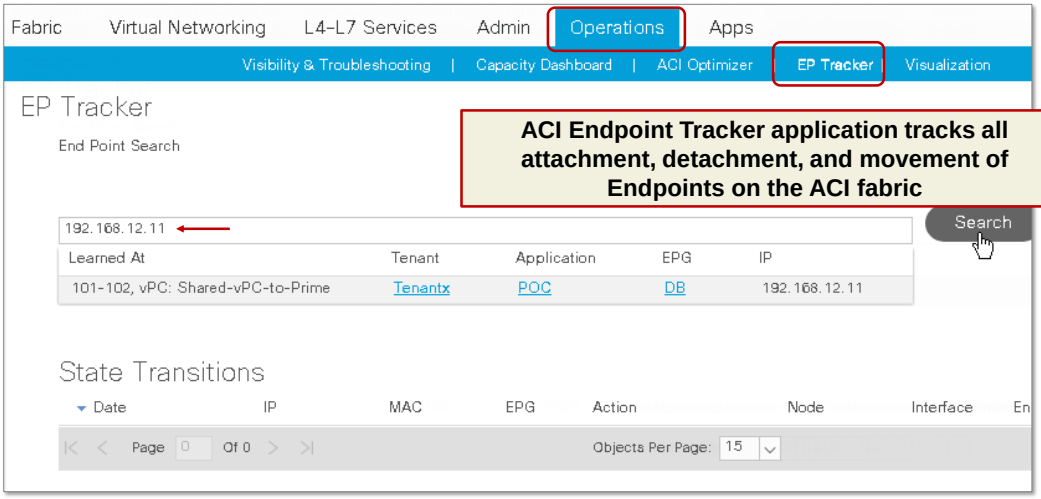
- Inspect End-point Operational status in Tenant
- Employ Endpoint Tracker
- View COOP Database
- Show endpoints on leaf and APIC
- Inspect EPG Faults
- Contract not configured/formed
- Fabric Access Policies
  - AEP / Domain / VLAN Pool
  - Interface / Switch Profiles
  - Interface Policies / Policy Groups



Common endpoint troubleshooting tasks:

- Inspect End-point Operational status in Tenant
- Employ Endpoint Tracker
- View COOP Database
- Show endpoints on leaf and APIC
- Inspect EPG Faults
- Contract not configured/formed
- Fabric Access Policies
  - AEP / Domain / VLAN Pool
  - Interface / Switch Profiles
  - Interface Policies / Policy Groups

EP Tracker in APIC Operations Tab



Endpoint (EP) Tracker

How is the EP Tracker used? EP Tracker enables you to view virtual and bare metal endpoint connections and disconnections to leaf switches and FEXes. Enter the IP or MAC address of an endpoint in the End Point Search field. Note that multiple endpoints can have the same MAC or IP address, which causes more than one endpoint to appear in the search results table. When an endpoint is selected from the search results, the connection and disconnection information appears in the State Transitions list. Disconnections are caused by aging out, the shutting down of a port, or when a server reboots. Connections and disconnections can be either in the same location or in another location if the endpoint was moved physically or virtually.

Layer 3 Interfaces:

The EP Tracker can search for ARP and neighbor discovery (ND) entries over Layer 3 interfaces. These entries can be searched using MAC, IPv4, or IPv6 addresses.
The following limitations apply:

- Searches using an ARP or ND entry through IPv4 or IPv6 addresses show all IPv4 and IPv6 entries with the same MAC address in the VRF column in the State Transitions list table.
- L3Out information is not listed.

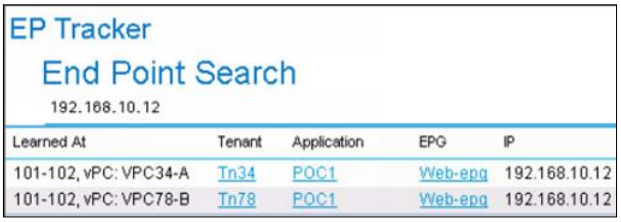
- Layer 3 interface information is not shown.
- The total number of all entries (endpoints, Layer 3 ARP, and Layer 3 IPv6 neighbor discovery entries) is limited to 8000.
- Layer 3 ARP and ND entries are not supported in the State Transitions list table.

End Point Search:
Depending on the type of endpoint you are searching for (regular or Layer 3), the search results appear in the table with slightly different information.

Lesson Exercise: EP Tracker

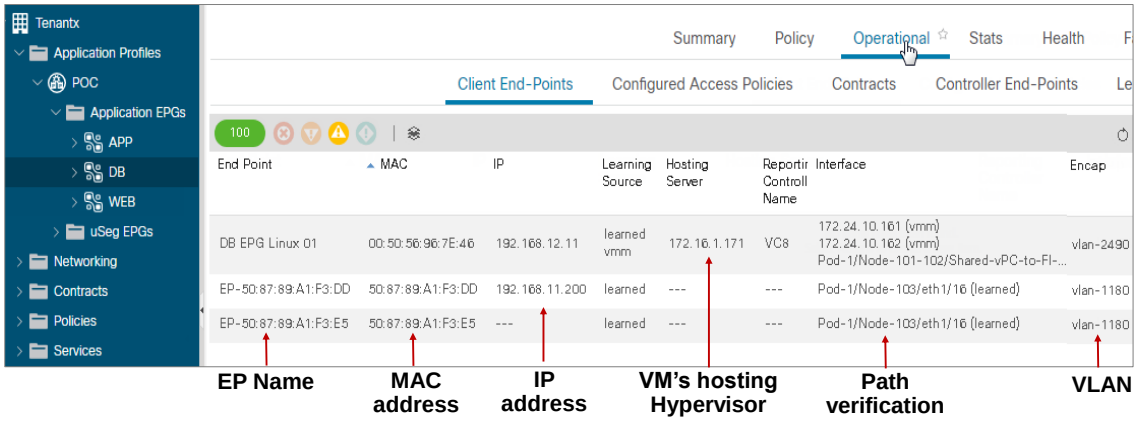
1. **Specify any virtual machine IP address**
2. **In APIC, go to: Operations | EP Tracker**
3. **Verify Results**

Note: All end-points with the same IP address will appear in the results table



End Point Verification Using EPG Operational Tab

Navigate to EPG OPERATIONAL Tab – verify End Point is learned, including path



Operational tab

This tab displays detailed information about the operational deployment and statistics between a managed object and its children.

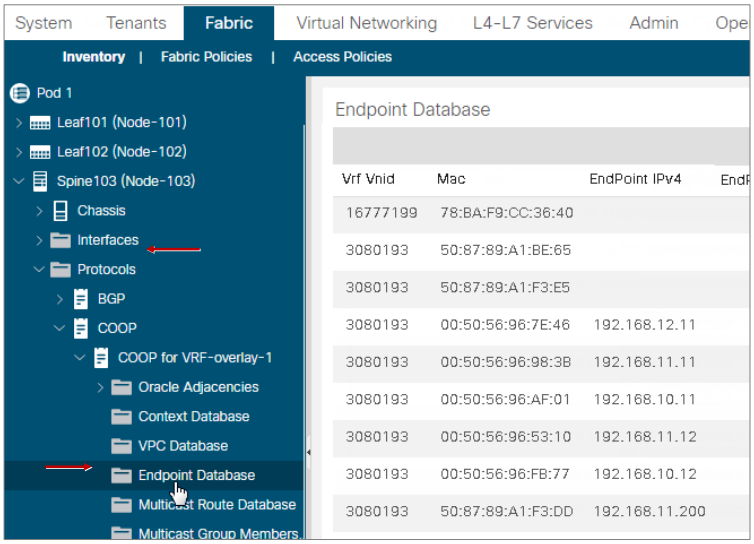
The information is displayed in the summary tables and other related sub-tabs.

- **End Point** - The physical or virtual client endpoint that requires services and policies.
- **MAC** - The MAC address for the client endpoint.
- **IP** - The IP address for the client endpoint
- **Learning Source** - The identifier for the learning source of the client endpoint. Possible values are:
- **vmm** - learned vmm
- **Hosting Server** - The identifier for the hosting server of the client endpoint.
- **Reporting Controller Name** - The reporting controller name of the client endpoint.
- **Interface** - The interface id assigned to the client endpoint.
- **Multicast Address** - The Multicast IP address for the client endpoint.
- **Encap** - The encapsulation (VLAN or VXLAN) of the virtual machine manager (VMM).

Viewing COOP Database

Fabric

- **Inventory**
  - **POD**
- Expand Spine
- Expand Protocols
- Expand COOP
- ***End Point Database***



Viewing Council Of Oracle Protocol (COOP) Database
The per-COOP instance information. There is only one instance of this object present in the system. Each leaf is responsible for reporting its local endpoints to the COOP database, located on each spine switch, which implies that all endpoint information in the Cisco ACI fabric is stored in the spine COOP database.

Because of spine proxy, Cisco ACI packet forwarding will work without remote endpoint learning. Spine proxy enables leaf switches to forward traffic directly to the COOP database located on the spine switches. Remote endpoint learning helps the ACI fabric forward packets more efficiently by allowing leaf switches to send packets directly to a destination leaf switch without using the resources on the spine switch that would be used to look up endpoints in the COOP database, which contains all the fabric endpoint information.

End Point Database – The endpoint database shows information about the endpoints in COOP. You can use this information to help debug the network state.

- Vrf Vnid – The network identifier for the VRF in the fabric.
- Mac – The MAC address of the endpoint.
- Endpoint IP – The IP address of the endpoint.

Displaying Endpoints on a Leaf

```
Leaf102# show endpoint
Legend:
 O - peer-attached    H - vtep-     a - locally-aged     S - static
 V - vpc-attached     p - peer-aged-L - local-    M - span
 s - static-arp       B - bounce
+---------------+--------------+-----------------+----------+-----------+
   VLAN/          Encap-          MAC Address          MAC Info/   Interface
   Domain         VLAN-           IP Address-IP Info
+---------------+--------------+-----------------+----------+-----------+
18/Tn34:VRF1     vxlan-15859678  547f.ee60.aa7c       p-         tunnel10
20--             vlan-2124       0050.5681.8c46       LV--       po4
Tn34:VRF1-       vlan-2124       192.168.11.11        LV
20--             vlan-2124       0050.5681.8218       LpV--      po4
Tn34:VRF1-       vlan-2124       192.168.11.12        LV
19--             vlan-2121       0050.56a5.4a73       LV--       po3
Tn34:VRF1-       vlan-2121       192.168.12.77        LV
30--             vlan-2147       0050.5681.854f       LV--       po8
Tn78:VRF1-       vlan-2147       192.168.11.11        LV
32--             vlan-2141       0050.5681.f366       LpV--      po8
```


You can verify discovered end point from both the leaf switch and APIC by invoking **show endpoint**.

The show endpoint command on any given leaf reports only endpoints learned on that leaf.

Displaying Endpoints Fabric-wide from APIC

```
apic1# show endpoints
Dynamic Endpoints:
Tenant      : Dev
Application : POC
AEPg-: APP
 End Point MAC      IP Address       Node      Interface      Encap       Multicast Address
 -----------------  ---------------  --------  -------------  ----------  ---------------
 00:50:56:AE:12:AF  192.168.11.11    101 102   vpc Dev-VPCB   vlan-2120   not-applicable
 00:50:56:AE:FD:45  192.168.11.12    101 102   vpc Dev-VPCB   vlan-2120   not-applicable

Tenant      : Dev
Application : POC
AEPg-: DB
 End Point MAC      IP Address       Node      Interface      Encap-Multicast Address
 -----------------  ---------------  --------  -------------  ----------  ---------------
 00:50:56:86:59:2D  192.168.12.100   101 102   vpc Dev-VPCB   vlan-2127    not-applicable

Tenant      : Dev
Application : POC
AEPg-: WEB
 End Point MAC      IP Address       Node      Interface      Encap       Multicast Address
 -----------------  ---------------  --------  -------------  ----------  ---------------
 00:50:56:AE:55:AB  ----       ---       ----    vlan-2124   not-applicable
 00:50:56:AE:5A:1E  192.168.10.11    101 102   vpc Dev-VPCA   vlan-2124   not-applicable
```


You can verify discovered end point from both the leaf switch and APIC by invoking **show endpoint**.

The show endpoint command on APIC reports endpoints fabric-wide.

Filter Tenant/EPG for a given IP address

Query for a given IP address & learn which Tenant & EPG it belongs:

- Output : Tenant, AEPs, MAC, VLAN and Leaf/Path information

```
apic1# show endpoints | grep -E "192.168.11.12|AEPg|Tenant|MAC"
```

```
apic1# show endpoint | grep -E "Tenant|EPg|192.168.10.12"
Tenant      : Tenant8
AEPg        : APP
Tenant      : Tenant8
AEPg        : DB
Tenant      : Tenant8
AEPg        : WEB
 00:50:56:96:FB:77  192.168.10.12                                    vlan-2450  not-applicable
 00:50:56:96:FB:77  192.168.10.12   101 102  vpc Shared-vPC-to-FI-B  vlan-2450  not-applicable
Tenant      : infra
AEPg        : default
```


This APIC command allows you to query for a particular IP address and determine which Tenant and EPG it belongs to.

```
apic1# show endpoints | grep -E
"192.168.11.12|AEPg|Tenant|MAC"
```

Troubleshoot Missing Endpoint Example

2. show endpoints – reveals end point is no longer visible in fabric

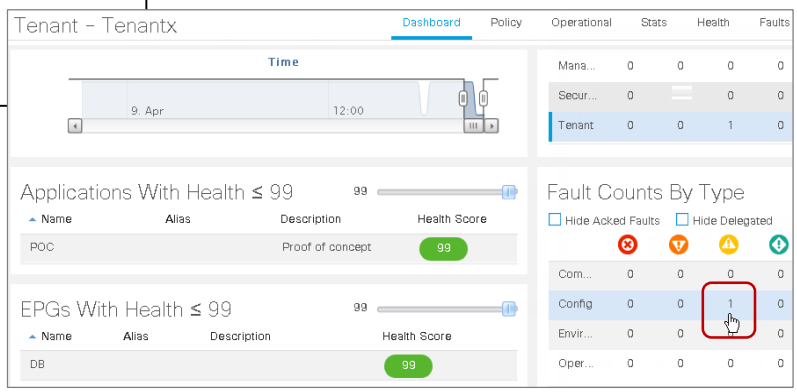
```
apic1# show endpoints ip 192.168.11.200
Legends:
(P):Primary VLAN
(S):Secondary VLAN

Total Dynamic Endpoints: 0
Total Static Endpoints: 0
```

1. Endpoint had become unreachable

```
From 192.168.11.200 icmp_seq=36673 Destination Host Unreachable
From 192.168.11.200 icmp_seq=36674 Destination Host Unreachable
```

3. Tenant Dashboard reports degraded Health score

- Minor fault appears in Fault table
- Double-click fault to drill-down

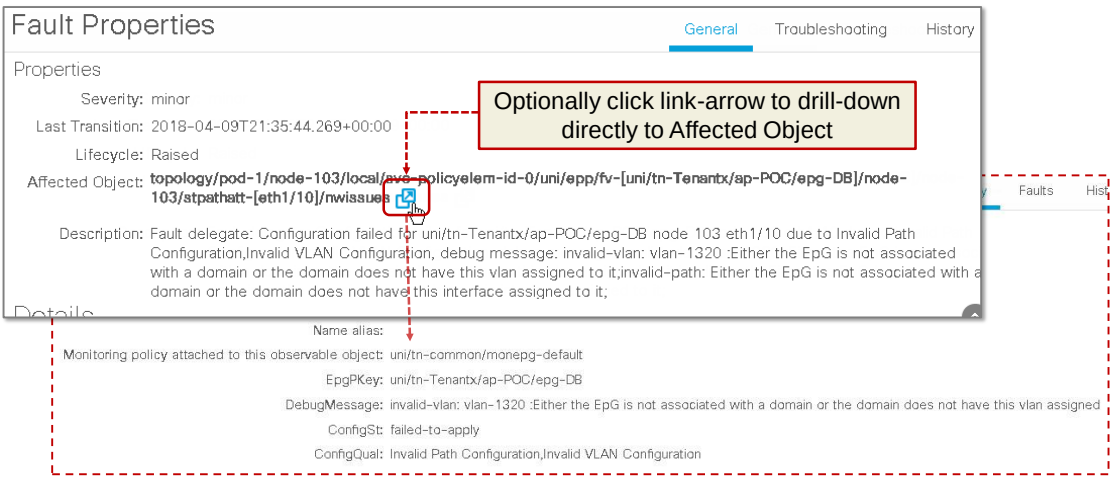


Troubleshoot Missing Endpoint Example

1. An Endpoint had become unreachable.
2. show endpoints – reveals end point is no longer visible in fabric.
3. Tenant Dashboard reports degraded Health score. A minor fault appears in Fault table. You can double-click on the fault to drill-down for detail.

Troubleshoot Missing Endpoint Example (cont.)

- Fault properties reveals invalid VLAN configured on static port assigned to EPG DB

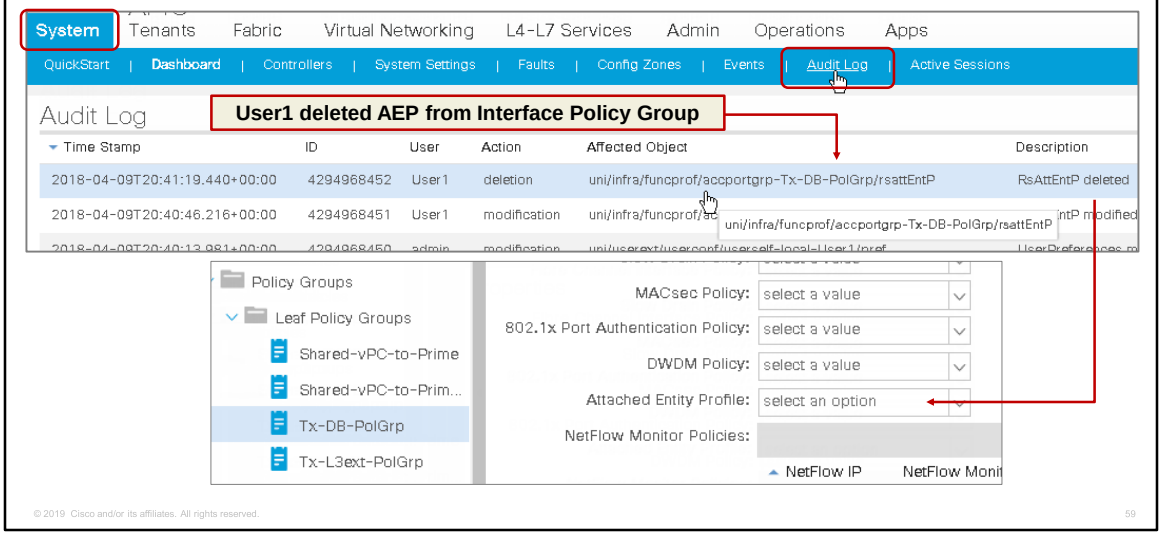


Troubleshoot Missing Endpoint Example

1. An Endpoint had become unreachable.
2. show endpoints – reveals end point is no longer visible in fabric.
3. Tenant Dashboard reports degraded Health score. A minor fault appears in Fault table. You can double-click on the fault to drill-down for detail.
4. Fault properties reveals invalid VLAN configured on static port assigned to EPG.

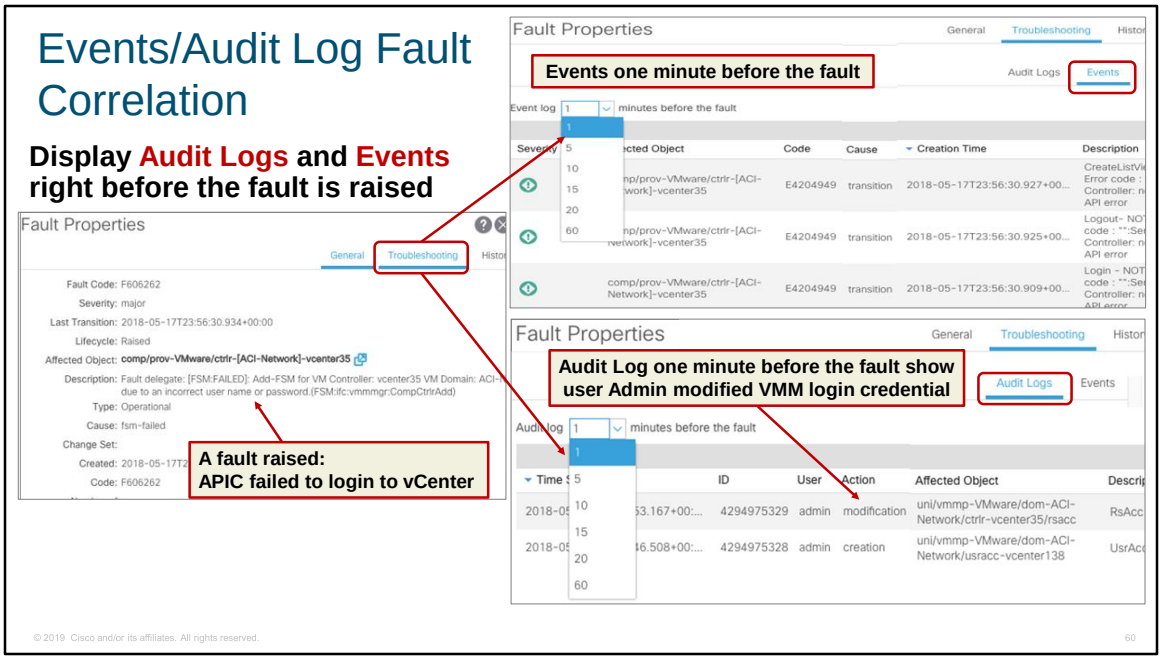
Troubleshoot Missing Endpoint Example (cont.)

- So.. **what changed**? Check **System Audit Log**



Troubleshoot Missing Endpoint Example

1. An Endpoint had become unreachable.
2. show endpoints – reveals end point is no longer visible in fabric.
3. Tenant Dashboard reports degraded Health score. A minor fault appears in Fault table. You can double-click on the fault to drill-down for detail.
4. Fault properties reveals invalid VLAN configured on static port assigned to EPG.
5. Since the application was previously operational, the indications are that a configuration change had occurred.
6. Checking the **System Audit Log** reports that User1 deleted the AEP from Interface Policy Group.

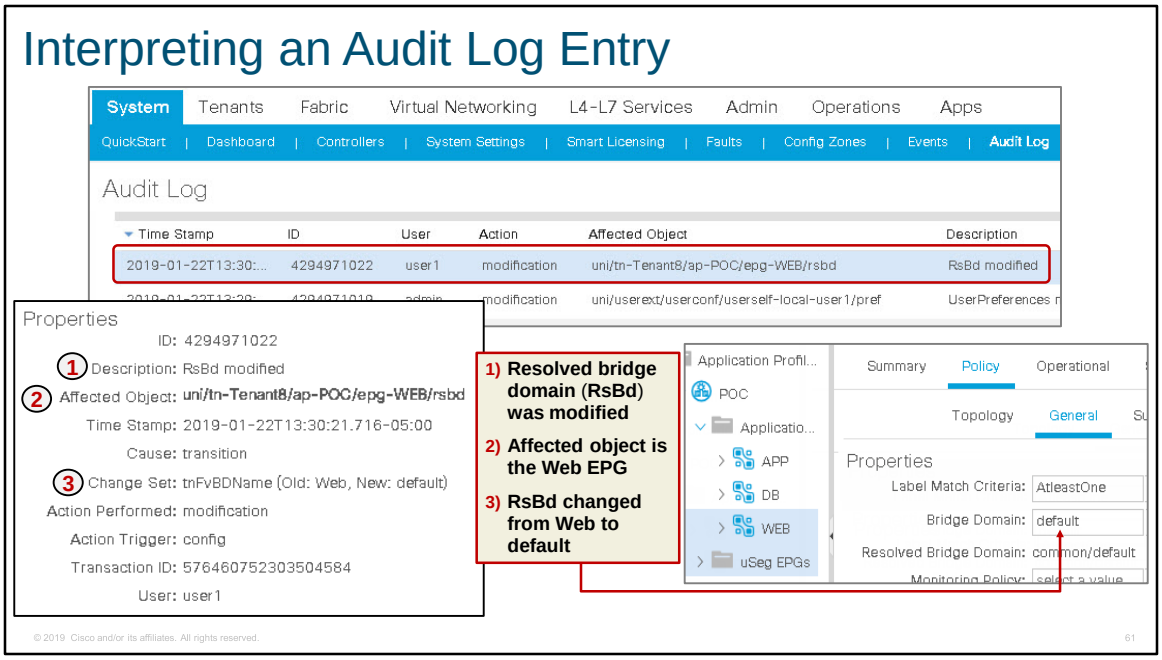


When troubleshooting a Fault, tracking changes can be done via the Events Log and Audit Log viewed from the Fault Properties.

- View the Event Log to report Affected Object, Cause, and Description. Filter the timeframe using the 'minutes before the fault' option.
- Use the Audit Log to identify what the specific change was made and who made the modification. Filter the timeframe using the 'minutes before the fault' option.

In the diagram, user admin modified the authentication on the VMM Domain ACI-Network.

Interpreting an Audit Log Entry

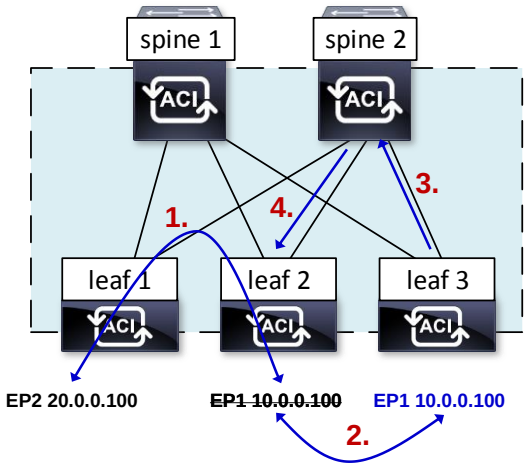


Audit records are objects that are created by the system to log user-initiated actions, such as login/logout and configuration changes. They contain the name of the user who is performing the action, a timestamp, a description of the action and, if applicable, the FQDN of the affected object. Audit records are never modified after creation and are deleted only when their number exceeds the maximum value specified in the audit retention policy.

Endpoint Move Scenarios

(DCACIO v4.0)

Reminder – How Moves are Handled

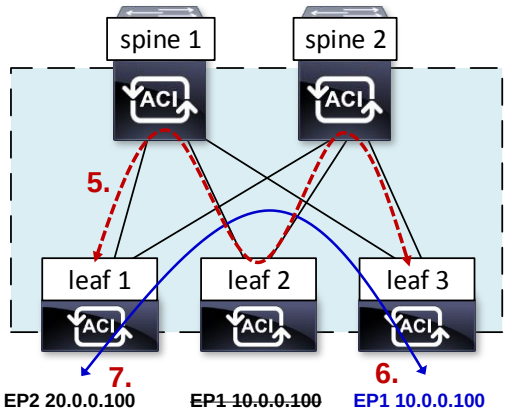


1. Dataplane traffic between EP1 and EP2 causes local and remote learns to be installed
2. EP1 moves to Leaf3
   - Local learn installed on Leaf3
3. Leaf3 sends COOP message to spine informing of new location
4. Spines install '**bounce**' flag to endpoint on Leaf2

How does the ACI fabric handle the movement of end-points across the fabric?

1. Dataplane traffic between EP1 and EP2 causes local and remote learns to be installed
2. EP1 moves to leaf 3. Local learn installed on leaf 3.
3. Leaf 3 sends COOP message to spine informing of new location
4. Spines install 'bounce' flag to ep on leaf 2 pointing to leaf 3
   - Bounce flag says "if I get traffic destined to this EP, I'm going to 'bounce' it to this tunnel for a period of time
   - During the bounce timer remote leafs should update EP information from dataplane traffic

Reminder – How Moves are Handled (cont.)



5. Leaf1 subsequently sends traffic to Leaf2
   - Leaf2 bounces packet to spine-proxy
   - Spine will forward to Leaf3
6. Leaf3 will send traffic directly to Leaf1
   - Leaf1 updates remote learn
7. Traffic now flows directly between Leaves 1 and 3

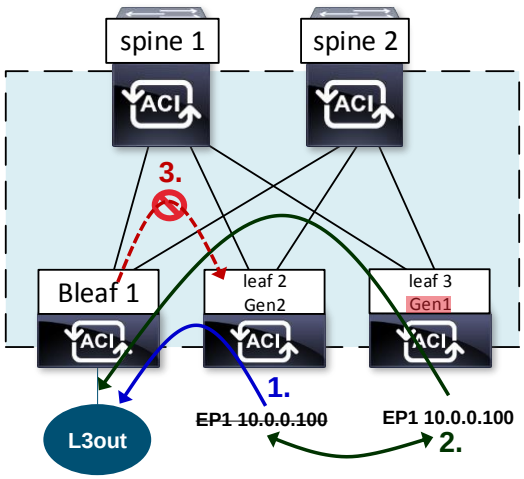
How does the ACI fabric handle the movement of end-points across the fabric?

1. Dataplane traffic between EP1 and EP2 causes local and remote learns to be installed
2. EP1 moves to leaf 3. Local learn installed on leaf 3.
3. Leaf 3 sends COOP message to spine informing of new location
4. Spines install 'bounce' flag to ep on leaf 2 pointing to leaf 3
   - Bounce flag says "if I get traffic destined to this EP, I'm going to 'bounce' it to this tunnel for a period of time
   - During the bounce timer remote leaves should update EP information from dataplane traffic
5. Leaf 1 sends traffic to leaf 2. Leaf 2 bounces to the spine-proxy, which then forwards to leaf 3.
6. Leaf 3 sends traffic directly to leaf 1. Leaf 1 updates remote learn.
7. Traffic now flows directly between leafs 1 and 3

Gen 1 and Gen 2 Mixed Fabric Problem

- Generation1 hardware: EP > L3out traffic does NOT trigger remote learn on border leaf (BL)
- Generation2 hardware: EP > L3out traffic will trigger remote learn on BL

1. **Leaf2 is Gen2** – BL remote learn installed
2. **Leaf3 is Gen1** – No remote learn on BL
   - EP1 > L3out traffic does refresh learn
3. **Leaf2** traffic is **black-holed** when **bounce flag** expires due to "stale" XR
   - Border Leaf1 still pointing to Leaf2



The exists a problem when ACI fabrics are deployed with a mix of generation 1 and generation 2 switch nodes.

- Generation 1 hardware: Endpoint > L3out traffic DOES NOT trigger remote learn on border leaf (BL)
- Generation 2 hardware: Endpoint > L3out traffic DOES trigger remote learn on BL

Scenario:

1. Leaf2 is Gen 2 so remote learn installed on Border Leaf1 upon endpoint move
2. Leaf3 is Gen 1 so remote learn on BL not moved. EP1 > L3out traffic does refresh learn
3. Once bounce flag expires on Leaf2 traffic is black-holed due to "stale" XR on Bleaf1 still pointing to Leaf2.

Troubleshooting Stale IP Remote Endpoints

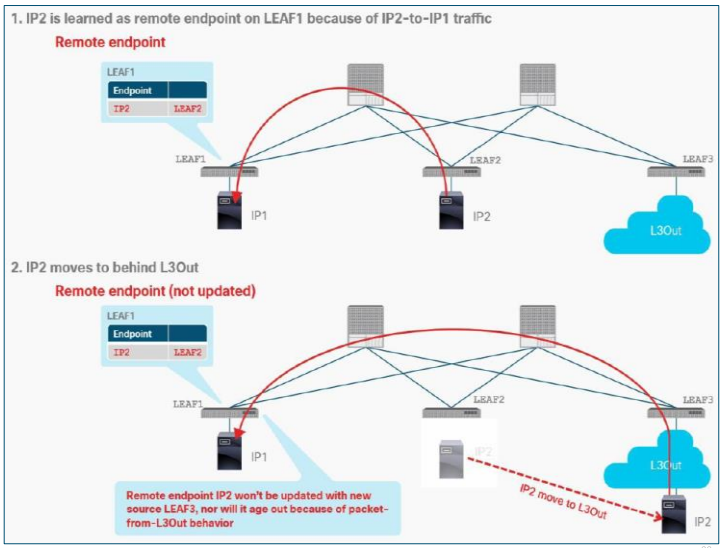
Scenario:

No source MAC or IP is learned as **new remote EP** by a packet

Consideration:

EP retention timer for existing remote EP is refreshed by this packet (from L3Out), even though other information is not updated (ie. originating leaf switch)

- May cause a **stale remote EP** to not age-out correctly after an EP is migrated to L3Out



66

Troubleshooting Stale IP Remote Endpoints

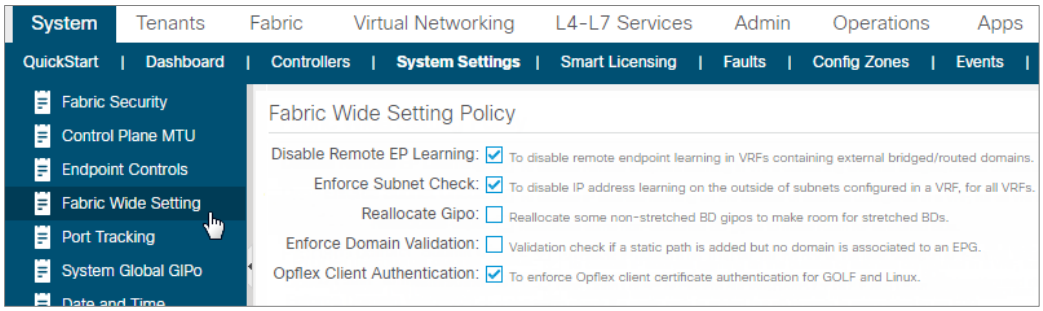
L3out behavior:

- Remote endpoint learning with an incoming packet from L3Out to Cisco ACI: No source MAC or IP address is learned as a new remote endpoint by a packet.

Consideration:

- The **endpoint retention timer** for an existing remote endpoint is refreshed by this packet from L3Out, even though other information, such as the originating leaf switch, is not updated.
- This behavior may cause a stale remote endpoint to not age-out correctly after an endpoint is migrated to L3Out from within Cisco ACI.

Stale IP Remote Endpoints – Policy Resolution



- **Disable Remote EP Learning**
- **Enforce Subnet Check**

Command to manually clear an endpoint by IP:

```
leaf# clear system internal epm endpoint key vrf <vrf-name> ip <ip-address>
```


Stale IP Remote Endpoints – Policy Resolution:

- Disable Remote Learning on BL's
  - Disables remote learning per-VRF on leafs with L3out deployed
- Enforce Subnet Check
  - Evaluates source IP against all subnets configured in BD's within VRF
  - Works for remote routed traffic sent on VRF vnid
  - Requires EX or later leafs
  - Best practice to enable

Manual CLI Resolution - Command to manually clear an endpoint by IP, run these commands on both directly connected leafs:

```
LEAF1# clear system internal epm endpoint key vrf <vrf-name> ip <ip-address>
```

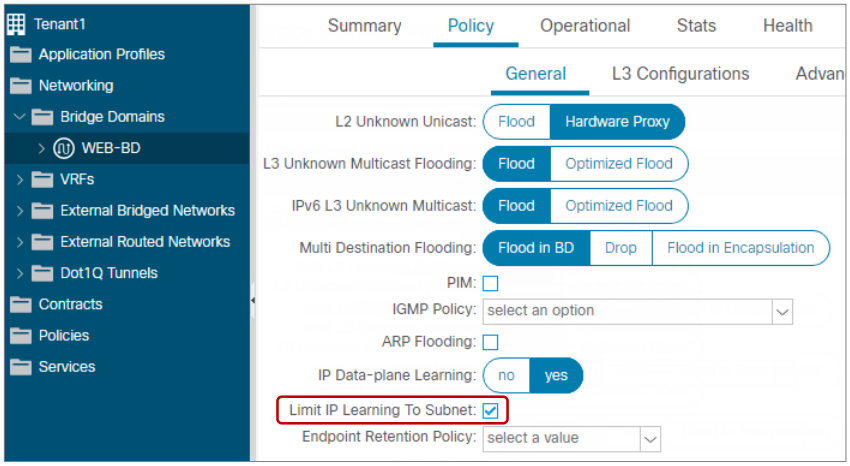
CSCvj17665 EP Announce support for stale IP remote endpoints - With this endpoint announcement feature, ACI will send an announcement message to all leaf switches when a bounce entry ages out to ensure the IP remote endpoints on the other leaf switches have the same information as the bounce entry and to delete the outdated IP remote endpoints, if any.

EPG Policy – Limit IP Learning To Subnet

EPG setting prevents endpoint learns if outside of BD subnet

- Mac is still learned, traffic still forwarded
- Always works for local learns
- Works for remote learns on L2 traffic sent to BD vnid
- Does not work for remote routed traffic sent on VRF vnid

Always enable for Layer 3 BD's



Another policy setting is available on any give EPG communicating with an outside L3out. Navigate to the EPG's Policy | General page.

- Verify **Limit IP Learning To Subnet** is checked

Benefits:

- Prevents endpoint learns if outside of BD subnet
- Mac is still learned, traffic still forwarded
- Always works for local learns

Works for remote learns on L2 traffic (sent to BD vnid)
Does not work for remote routed traffic sent on VRF vnid
Always enable for Layer 3 BD's

Endpoint Learning Summary

- Ensure "**Limit IP Learning to Subnet**" is enabled on L3 BD's
- Enable "**Enforce Subnet Check**"
- Enable "**Disable Remote EP Learning**" on BL's in mixed fabrics
- Upgrade to 3.2(2) or later for EP Announce on bounce deletion

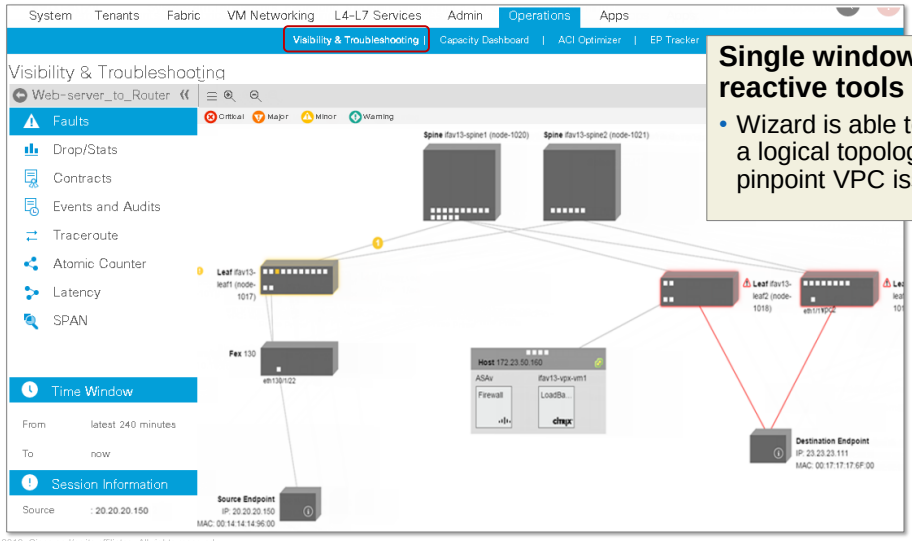
Endpoint Learning Summary

- Ensure "Limit IP Learning to Subnet" is enabled on L3 BD's
- Enable "Enforce Subnet Check"
- Enable "Disable Remote EP Learning on BL's" in mixed fabrics
- Upgrade to 3.2(2) or later for EP Announce on bounce deletion
- For other features and scenarios: Endpoint Learning White Paper

Visibility & Troubleshooting Tool

(DCACIO v4.0)

Visibility & Troubleshooting Tool



Single window to multiple reactive tools

- Wizard is able to quickly draw a logical topology, as well as pinpoint VPC issue

Using the Troubleshooting Wizard

The Troubleshooting Wizard allows you understand and visualize how your network is behaving, which can ease your networking concerns should issues arise.

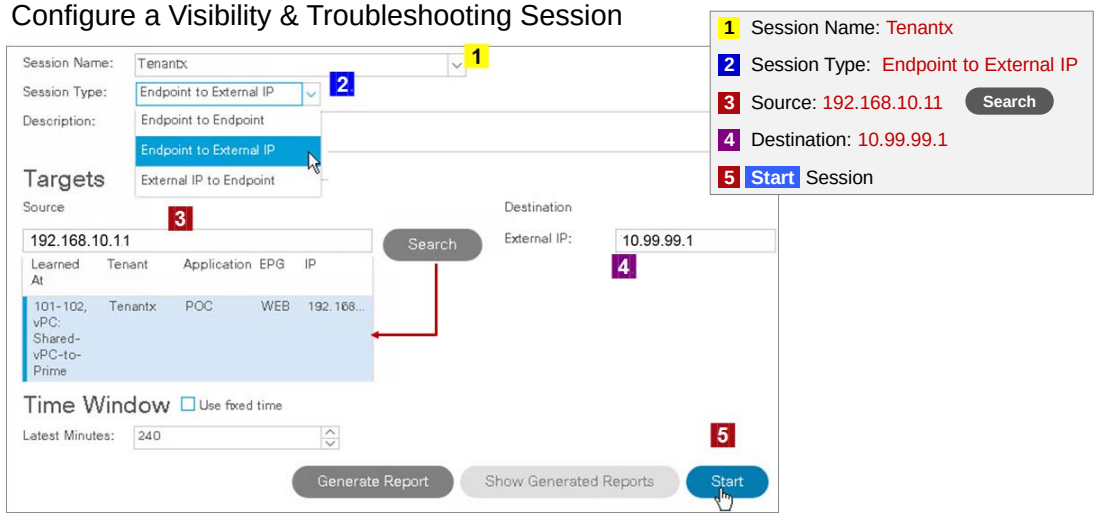
This wizard allows Administraters to troubleshoot issues that occur during specific time frames, which can be designated by selecting two endpoints. For example, you may have two endpoints that are having intermittent packet loss but you don't understand why. Through the troubleshooting GUI, you can evaluate the issue so that you can effectively resolve it rather than logging onto each machine that you suspect to be causing this faulty behavior.

Since you may want to revisit the session later, you should give the session a unique name. You may also choose to use a pre-configured test. You can debug from endpoint to endpoint, or from an internal or external endpoint, or from an external to an internal endpoint.

Further, you can define a time window in which you want to perform the debug. The Troubleshooting GUI allows you to enter a source and destination endpoint for the endpoints you are looking for. You can do this with a MAC, IPv4, or IPv6 address and then select by tenant. You also have the option to generate a troubleshooting report that can be sent to TAC.

Launching a Visibility/Troubleshooting Wizard

- Configure a Visibility & Troubleshooting Session



Each student will launch a Visibility & Troubleshooting session accordingly.

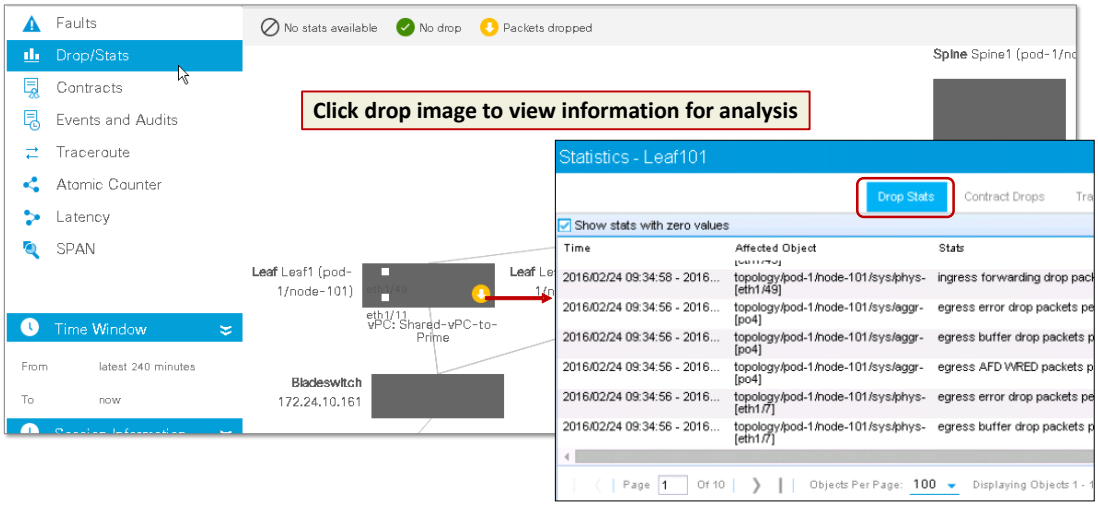
Launching a Visibility/Troubleshooting Wizard – Before you start using the Troubleshooting Wizard, you must be logged on as an Administrative user. Then you must designate Source and Destination endpoints (Eps) and select a time window for your troubleshooting session. The time window is used for retrieving Events, All Records, Deployment Records, Audit Logs, and Statistics. (You can also set or modify the Description and Time Window from the left navigation pane of the Troubleshooting Wizard at any time.)

Since you may want to revisit the session later, you should give the session a unique name. You may also choose to use a pre-configured test. You can debug from endpoint to endpoint, or from an internal or external endpoint, or from an external to an internal endpoint.

Further, you can define a time window in which you want to perform the debug. The Troubleshooting GUI allows you to enter a source and destination endpoint for the endpoints you are looking for. You can do this with a MAC, IPv4, or IPv6 address and then select by tenant. You also have the option to

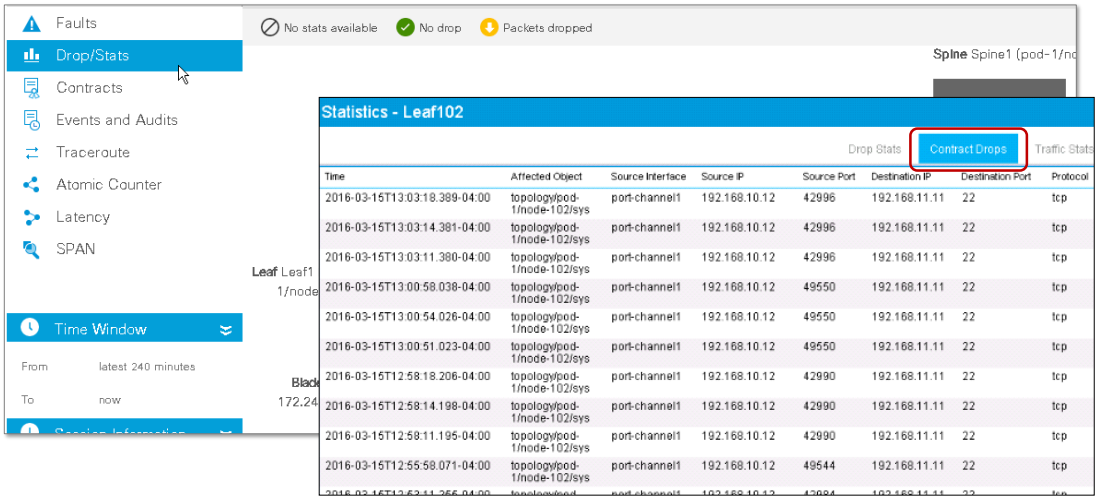
generate a troubleshooting report that can be sent to TAC.

Visibility & Troubleshooting – Drops/Stats



Drop/Starts – Displays all the statistics from the faults so that you can clearly see where drops exist or not. You can click on any drop image to see more information for analysis.

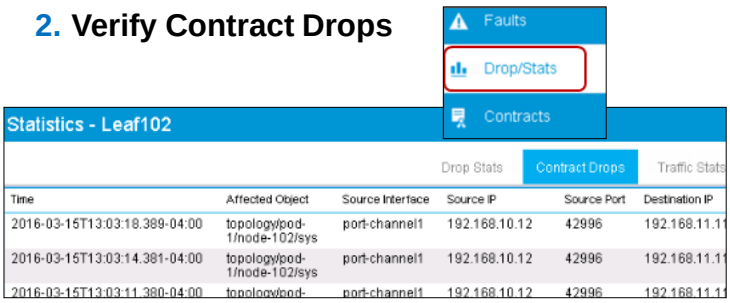
Visibility & Troubleshooting – Contract Drops

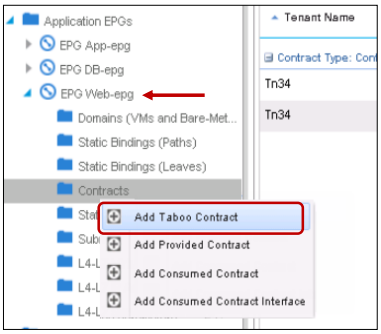


Contract Drops – Displays all packets dropped that failed to match contract filters.

Lesson exercise: Viewing Contract Drops

1. **In your tenant, re-apply Taboo contract to Web EPG**
2. **Verify Contract Drops**



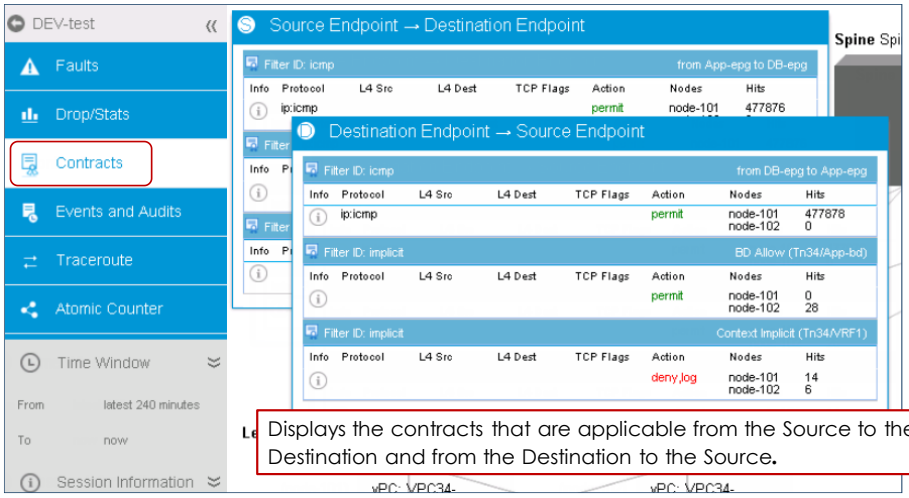


Lesson exercise: Viewing Contract Drops

In this Lesson Exercise you will view Contract Drops in the Visibility & Troubleshooting session by7 re-applying the Taboo contract.

1. In your tenant, re-apply Taboo contract to Web EPG
2. Verify Contract Drops

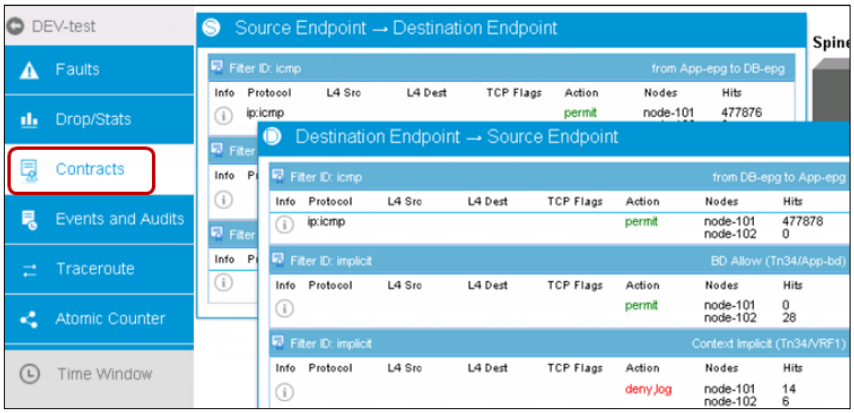
Visibility & Troubleshooting – Contracts



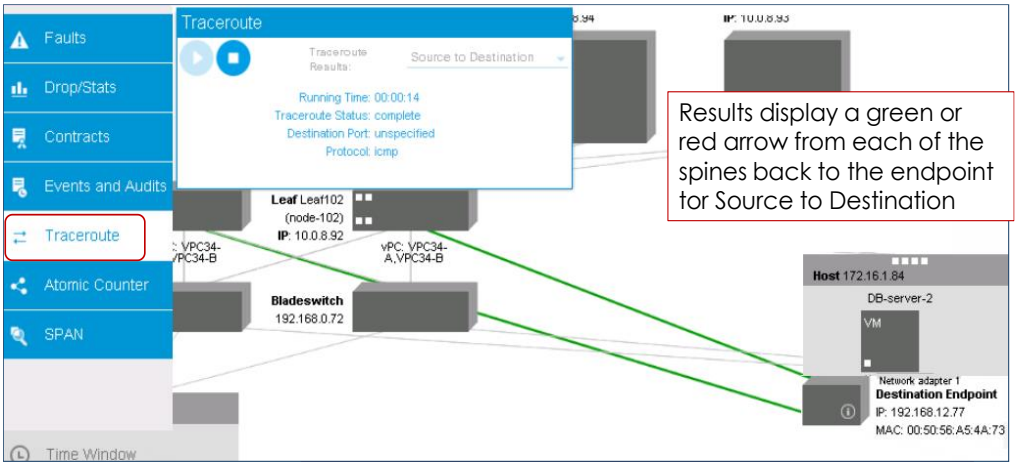
Contracts – Displays the contracts that are applicable from the Source to the Destination and from the Destination to the Source.

Lesson exercise: Viewing Contracts

View Contracts in current Visibility & Troubleshooting session



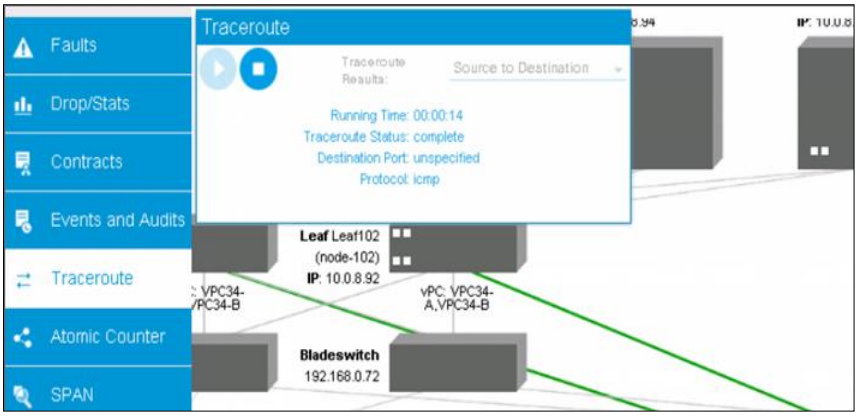
Visibility & Troubleshooting – Traceroute



Traceroute – Create and run a traceroute. Once the traceroute completes, you can see where it was launched and what the result was. The results display a green or red arrow from each of the spines back to the endpoint tor Source to Destination.

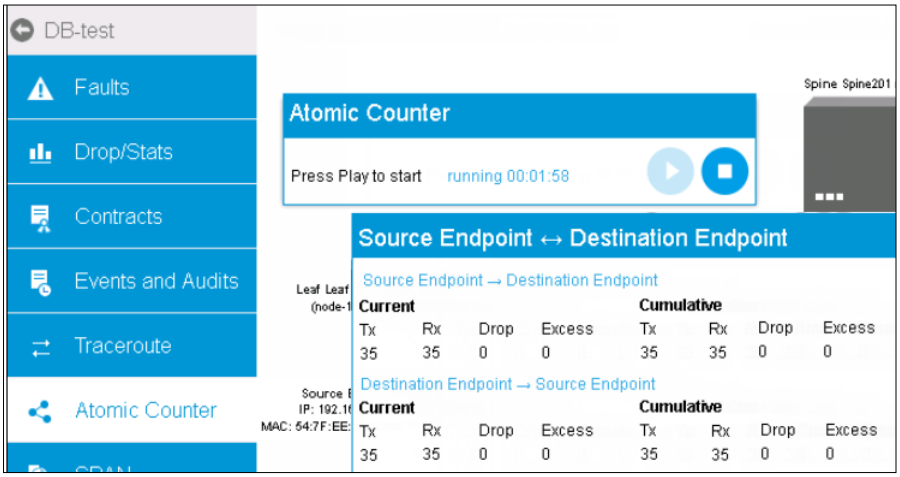
Lesson Exercise: Traceroute

Invoke Traceroute in current Visibility & Troubleshooting session



Visibility & Troubleshooting – Atomic Counters

Atomic counters require NTP to be enabled on the fabric



Atomic Counters are useful for troubleshooting connectivity between endpoints, EPGs, or an application within the fabric. A user reporting application may be experiencing slowness, or atomic counters may be needed for monitoring any traffic loss between two endpoints. One capability provided by atomic counters is the ability to place a trouble ticket into a proactive monitoring mode, for example when the problem is intermittent, and not necessarily happening at the time the operator is actively working the ticket.

Atomic counters can help detect packet loss in the fabric and allow the quick isolation of the source of connectivity issues. Atomic counters require NTP to be enabled on the fabric.

Leaf-to-leaf (TEP to TEP) atomic counters can provide the following:

- Counts of drops, admits, and excess packets
- Short-term data collection such as the last 30 seconds, and long-term data collection such as 5 minutes, 15 minutes, or more
- A breakdown of per-spine traffic (available when the number of TEPs, leaf or VPC, is less than 64)
- Ongoing monitoring

Leaf-to-leaf (TEP to TEP) atomic counters are cumulative and cannot be cleared.

However, because 30 second atomic counters reset at 30 second intervals, they can be used to isolate intermittent or recurring problems.
Tenant atomic counters can provide the following:

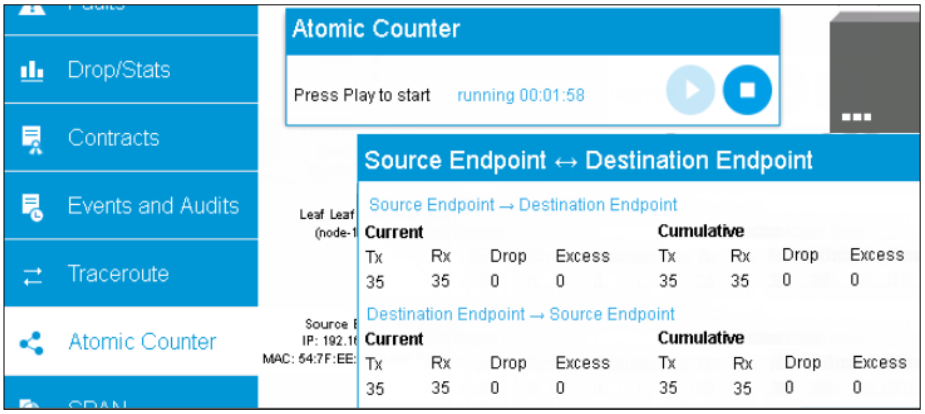
- Application-specific counters for traffic across the fabric, including drops, admits, and excess packets
- Modes include the following:
- Endpoint to endpoint MAC address, or endpoint to endpoint IP address. Note that a single target endpoint could have multiple IP addresses associated with it.
- EPG to EPG with optional drill down
- EPG to endpoint
- EPG to * (any)
- Endpoint to external IP address

Atomic counters track the amount packets of between the two endpoints and use this as a measurement. They do not take into account drops or error counters in a hardware level.

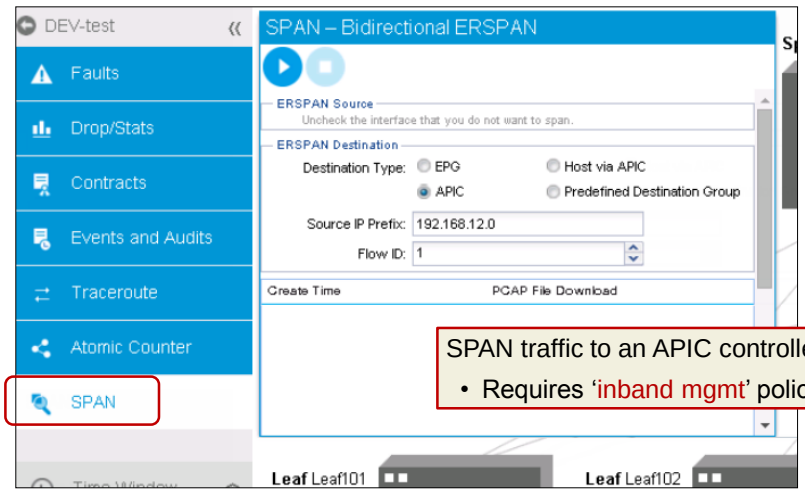
- Dropped packets are calculated when there are less packets received by the destination than transmitted by the source.
- Excess packets are calculated when there are more packets received by the destination than transmitted by the source.

Lesson Exercise: Atomic Counters

Invoke Atomic Counters in current Configure a Visibility & Troubleshooting session



Visibility & Troubleshooting – SPAN



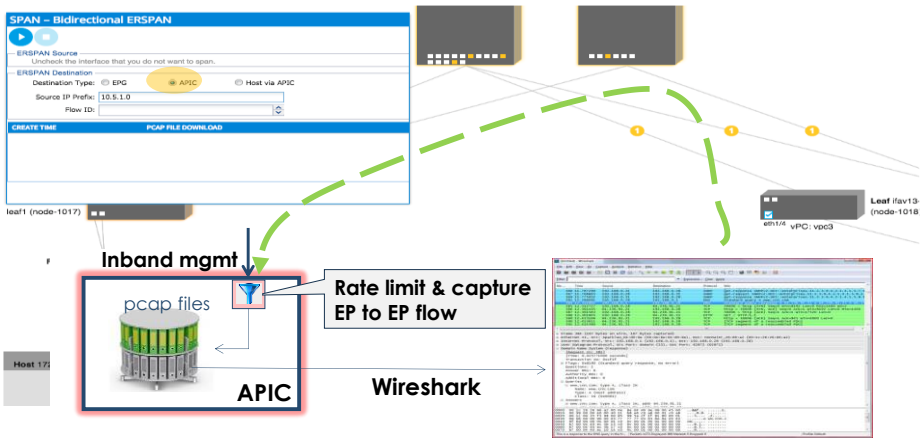
SPAN bi-directional traffic and redirect it to an analyzer

SPAN traffic to an APIC controller will be throttled
- Requires 'inband mgmt' policy

SPAN – Used to span (or mirror) bi-directional traffic and redirect it to the analyzer. In a SPAN, you are making a copy and sending it to a SPAN destination where a packet analyzer tool is available. Following options are available when you configure this SPAN destination:

- EPG—SPAN traffic is sent to an analyzer available in an EPG within the ACI fabric.
- APIC Controller—SPAN traffic is sent to an APIC controller where packets are stored in a file that can be downloaded and viewed. Requires inband mgmt policy. Note: SPAN traffic to an APIC controller will be throttled.
- Host reachable via APIC controller—SPAN traffic is sent to a host (VM/server) that is reachable via APIC controller and has a packet analyzer tool (i.e. wireshark) installed. Note: SPAN traffic to an APIC controller will be throttled.
- Predefined destination group—SPAN traffic is sent to an existing (predefined) SPAN destination group.

Visibility & Troubleshooting – SPAN to APIC



[Note: The presenter has included additional notes which are located below the transcript text.]

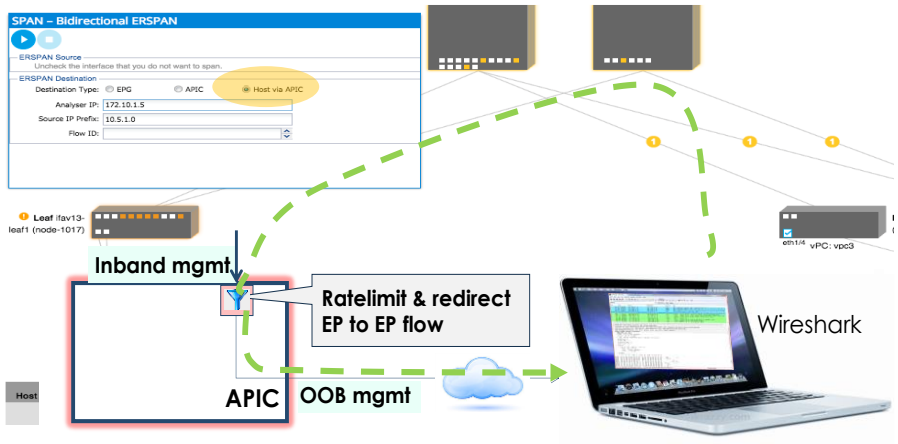
So this is the APIC part that I was talking about. So once you pick APIC the packets are actually sent from here out to the spine and back to the APIC and then stored in PCAP files essentially. We rate limit it so that there is no bombarding of APIC CPU.

Also one of the CM parts that is required for this functionality is the in-band management. So in-band management has to be turned on. Essentially, that's the only way the switch spans the packets. It cannot span out-of-band, so it does span in-band and then in-band management is a requirement for this functionality.

Additional Presenter Notes:

In-band mgmt needs to be configured

Visibility & Troubleshooting – SPAN to Laptop



The next step essentially I was talking about is SPAN to laptop. So when you say forced via APIC, so it actually goes to the APIC and then goes to the host itself. Again, rate limited on the APIC. You'll see only a fraction of the packets.

Troubleshooting Access Ports

(DCACIO v4.0)

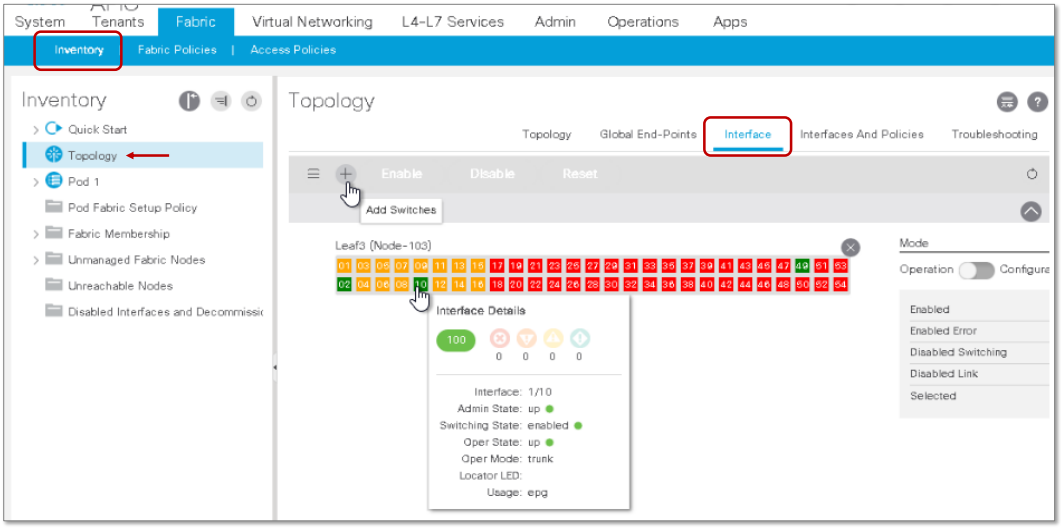
Unicast Data Plane Forwarding and Reachability

Unicast forwarding and reachability problems can be, but is not limited to:

- End points not showing up in the forwarding tables
- End points not able to communicate with each other; non zoning rule policy (contract) related problems
- VLANs not being programmed
- Incorrect configurations causing these problems and subsequent faults raised

This section covers various unicast forwarding and reachability problems. This can be, but is not limited to, end points not showing up in the forwarding tables, end points not able to communicate with each other (non zoning rule policy (contract) related problems), VLANs not being programmed, as well as incorrect configurations that can cause these problems and the subsequent faults that are raised.

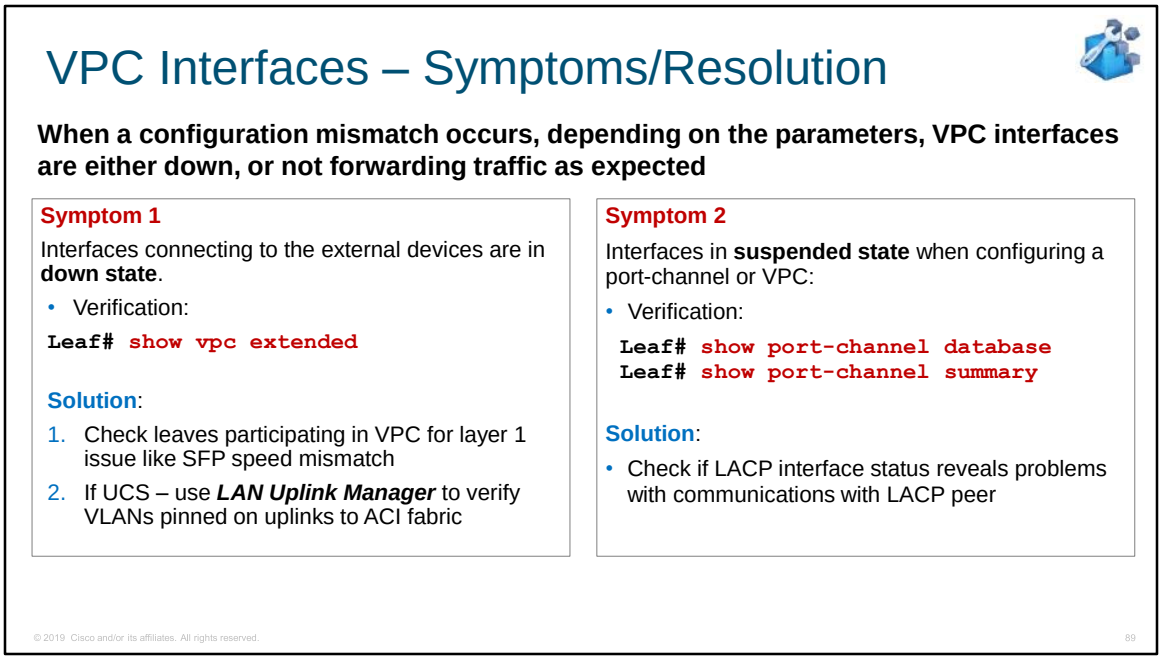
Topology View – Verify Port Status



CLI – Verify Port Status

```
Leaf103# show interface brief | more
--------------------------------------------------------------------------------
Port   VRF      Status IP Address                              Speed      MTU
--------------------------------------------------------------------------------
mgmt0  --       up     172.16.1.18                             1000       9000
--------------------------------------------------------------------------------
Ethernet      VLAN    Type Mode    Status Reason               Speed     Port
Interface                                                                Ch #
--------------------------------------------------------------------------------
Eth1/1        0       eth  trunk   up     none                 10G(D)    --
Eth1/2        0       eth  trunk   up     none                 10G(D)    --
Eth1/3        0       eth  trunk   up     none                 10G(D)    --
Eth1/4        0       eth  trunk   up     out-of-service       10G(D)    --
Eth1/5        0       eth  trunk   up     none                 10G(D)    --
Eth1/6        0       eth  trunk   up     out-of-service       10G(D)    --
Eth1/7        0       eth  trunk   up     out-of-service       10G(D)    --
Eth1/8        0       eth  trunk   up     none                 10G(D)    --
Eth1/9        0       eth  trunk   up     out-of-service       1000(D)   --
Eth1/10       0       eth  trunk   up     out-of-service       1000(D)   --
```


show interface – command has not gone away



VPC Interfaces – Symptoms/Resolution

When a configuration mismatch occurs, depending on the parameters, VPC interfaces are either down, or not forwarding traffic as expected.

Common issues with virtual port-channels include:

- Interfaces connecting to the external devices are in down state.
- Interfaces in suspended state when configuring a port-channel or VPC

This diagram details the resolution process.

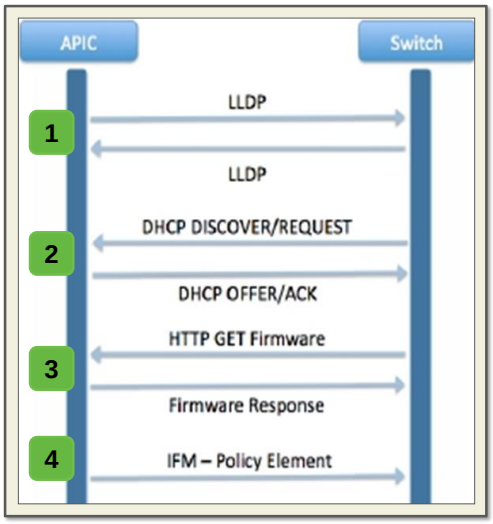
Troubleshooting the Fabric Discovery Process (LLDP)

(DCACIO v4.0)

Fabric Discovery Process

- ACI fabric is brought up in a cascading manner, starting with leaf node nodes directly attached to the APIC
- LLDP and control-plane IS-IS convergence occurs in parallel to this boot process

1. LLDP Neighbor Discovered
2. TEP IP address assigned to node
3. Node software upgraded
4. Policy Element IFM (Intra-Fabric Messaging) Setup



Startup Discovery and Configuration – The clustered APIC controller provides DHCP, bootstrap configuration, and image management to the fabric for automated startup and upgrades.

The ACI fabric bootstrap sequence begins when the fabric is booted with factory-installed images on all the switches. The Cisco Nexus 9000 Series switches that run the ACI firmware and APICs use a reserved overlay for the boot process. This infrastructure space is hard-coded on the switches. The APIC can connect to a leaf through the default overlay, or it can use a locally significant identifier.

The ACI fabric uses an infrastructure space, which is securely isolated in the fabric and is where all the topology discovery, fabric management, and infrastructure addressing is performed. ACI fabric management communication within the fabric takes place in the infrastructure space through internal private IP addresses. This addressing scheme allows the APIC to communicate with fabric nodes and other Cisco APIC controllers in the cluster. The APIC discovers the IP address and node information of other Cisco APIC controllers in the cluster using the Link Layer Discovery Protocol (LLDP)-based discovery process.

The following describes the APIC cluster discovery process:

- Each APIC in the Cisco ACI uses an internal private IP address to communicate with the ACI nodes and other APICs in the cluster. The APIC discovers the IP address of

other APIC controllers in the cluster through the LLDP-based discovery process.

- APICs maintain an **appliance vector** (**AV**), which provides a mapping from an APIC ID to an APIC IP address and a universally unique identifier (UUID) of the APIC. Initially, each APIC starts with an AV filled with its local IP address, and all other APIC slots are marked as unknown.
- When a switch reboots, the policy element (PE) on the leaf gets its AV from the APIC. The switch then advertises this AV to all of its neighbors and reports any discrepancies between its local AV and neighbors' AVs to all the APICs in its local AV.

Using this process, the APIC learns about the other APIC controllers in the ACI through switches. After validating these newly discovered APIC controllers in the cluster, the APIC controllers update their local AV and program the switches with the new AV. Switches then start advertising this new AV. This process continues until all the switches have the identical AV and all APIC controllers know the IP address of all the other APIC controllers.

The ACI fabric is brought up in a cascading manner, starting with the leaf node nodes that are directly attached to the APIC. LLDP and control-plane IS-IS convergence occurs in parallel to this boot process. The ACI fabric uses LLDP- and DHCP-based fabric discovery to automatically discover the fabric switch nodes, assign the infrastructure VXLAN tunnel endpoint (VTEP) addresses, and install the firmware on the switches. Prior to this automated process, a minimal bootstrap configuration must be performed on the Cisco APIC controller.
After the APIC controllers are connected and their IP addresses assigned, the APIC GUI can be accessed by entering the address of any APIC controller into a web browser. The APIC GUI runs HTML5 and eliminates the need for Java to be installed locally.

In this discovery process, a fabric node is considered active when the APIC and node can exchange heartbeats through the **Intra-Fabric Messaging** (**IFM**) process. The IFM process is also used by the APIC to push policy to the fabric leaf nodes.
Fabric discovery happens in three stages. The leaf node directly connected to the APIC is discovered in the first stage. The second stage of discovery brings in the spines connected
to that initial seed leaf. Then the third stage processes the discovery of the other leaf nodes and APICs in the cluster.
The diagram below illustrates the discovery process for switches that are directly connected
to the APIC. Coverage of specific verification for other parts of the process will be presented later in the chapter.
The steps are:

- Link Layer Discovery Protocol (LLDP) Neighbor Discovery
- Tunnel End Point (TEP) IP address assignment to the node
- Node software upgraded if necessary
- Policy Element IFM Setup

Node status may fluctuate between several states during the fabric registration process. The states are shown in the Fabric Node Vector table. The APIC CLI command to show the Fabric Node Vector table acidiag fnvread and sample output will be shown further down in this section. Below is a description of each state.

- Unknown – Node discovered but no Node ID policy configured
- Undiscovered – Node ID configured but not yet discovered
- Discovering – Node discovered but IP not yet assigned
- Unsupported – Node is not a supported model
- Disabled – Node has been decommissioned
- Inactive – No IP connectivity
- Active – Node is active

Leaf CLI Verification

From the leaf CLI, invoke: **show lldp neighbor**

- Essential in determining whether an APIC lldp is coming from the APIC OS or the VIC

Good!

```
leaf1# show lldp neighbors
Capability codes:
  (R) Router, (B) Bridge, (T) Telephone, (C) DOCSIS Cable Device
  (W) WLAN Access Point, (P) Repeater, (S) Station, (O) Other
Device ID       Local Intf      Hold-time  Capability  Port ID
apic1           Eth1/46         120                    eth2-1
```

Bad!

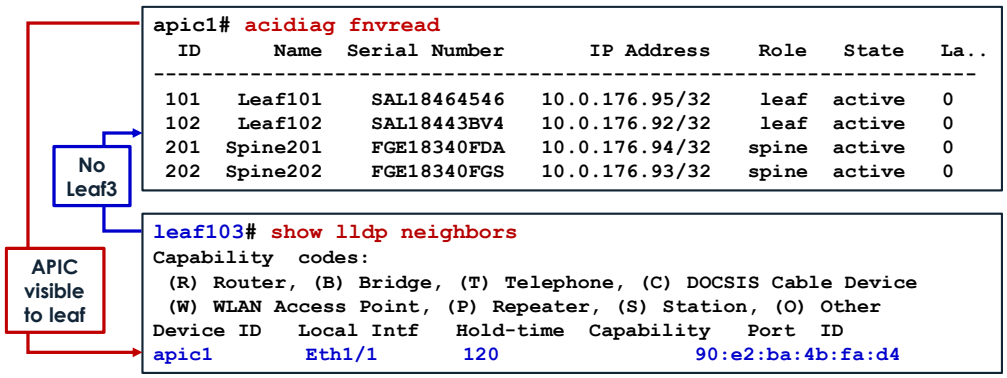
```
leaf1# show lldp neighbors
Capability codes:
  (R) Router, (B) Bridge, (T) Telephone, (C) DOCSIS Cable Device
  (W) WLAN Access Point, (P) Repeater, (S) Station, (O) Other
Device ID       Local Intf      Hold-time  Capability  Port ID
                Eth1/46         120                    eth2-1
```


”show lldp neighbors” from the leaf is essential in determining whether an APIC lldp is coming from the APIC OS or the VIC

If there is an entry but the “Device ID” is not showing the hostname of the APIC, the VIC may be configured with LLDP enabled

Common Fabric Bring-Up Issue

- APIC does **not** display the leaf in the output of "**`acidiag fnvread`**"
- Leaf disaplays the APIC as LLDP neighbor "**`show lldp neighbors`**"



Common issues seen when bringing up the initial hardware.
The APIC Fabric can be ordered in several different configurations. There is an option to purchase optical leaves (leaves with Small Form Pluggable (SFP) interfaces), and when that is the case an optical Virtual Interface Card (VIC1225) must be used in the APIC. When a copper leaf is used the optical VIC1225T must be used.
Initial cabling of the ACI fabric is very important and the following requirements must be adhered to:

- Leafs can only be connected to spines. There should be no cabling between the leafs.
- Spines can only be connected to leafs. Spines can not be inter-connected.
- An APIC must be attached to a leaf. APICs should be dual-homed (connected to two different leafs) for redundancy.
- All end points, L2, L3, L4-L7 devices must connect to leafs. Nothing should be connected to spines other than leafs as previously mentioned

Problem Description
There are a few common issues that can be observed when initially bringing up a fabric.
Symptom 1 – On the connection between the APIC and leaf, the APIC side is down

(no lights) but the leaf side has lights on.
Verification/Resolution:

- The leaf showed the APIC as a LLDP neighbor (show lldp neighbors)
- The APIC did not show the leaf in the output of "acidiag fnvread"
- A physical examination of the setup shows:
    - In the picture above a GLC-T transceiver was plugged into the APIC which has a VIC1225 installed. This is an optical SFP+ Virtual Interface Card (VIC). The other end of the connection is the 93128TX (copper) leaf.

- Their desired behavior was to convert optical to copper (media conversion).
- There are no transceivers qualified to do this sort of conversion. Optical-based VICs need to be plugged into optical-based leafs, and copper based VICs need to be plugged into copper-based leafs.

Once the proper transceiver was used, and a leaf with copper ports was connected the other end, the link came up properly and both the APIC and the leaf were able to share LLDP as expected. Fabric discovery was able to continue as expected.

Common Fabric Bring-Up Issue – Resolution

Physical examination reveals GLC-T transceiver was plugged into APIC VIC1225 optical SFP+

- The other end of the connection is the 93128TX (copper) leaf
- Install proper transceiver and verify


```
apic1# acidiag fnvread
  ID       Name   Serial Number       IP Address     Role    State
LastUpdMsgId
--------------------------------------------------------------------------
 101    Leaf101     SAL18464546    10.0.176.95/32     leaf   active    0
 102    Leaf102     SAL18443BV4    10.0.176.92/32     leaf   active    0
 103    Leaf103     SAL1948U35D    10.0.176.91/32     leaf   active    0
 201   Spine201     FGE18340FDA    10.0.176.94/32    spine   active    0
 202   Spine202     FGE18340FGS    10.0.176.93/32    spine   active    0
```


Verification/Resolution:

- A physical examination of the setup shows:
  - In the picture above a GLC-T transceiver was plugged into the APIC which has a VIC1225 installed. This is an optical SFP+ Virtual Interface Card (VIC). The other end of the connection is the 93128TX (copper) leaf.
- Their desired behavior was to convert optical to copper (media conversion).
- There are no transceivers qualified to do this sort of conversion. Optical-based VICs need to be plugged into optical-based leafs, and copper based VICs need to be plugged into copper-based leafs.

Once the proper transceiver was used, and a leaf with copper ports was connected the other end, the link came up properly and both the APIC and the leaf were able to share LLDP as expected. Fabric discovery was able to continue as expected.

Hardware Diagnostics and Replacement

(DCACIO v4.0)

Identify Hardware Failure

Examples of hardware events that generate syslog messages and SNMP traps include:

- **Linecard failure on a spine switch**
- **Supervisor failure on a spine switch**
- **System controller failure on a spine switch**
- **Power supply or fan failures on a leaf or a spine switch**

Identify Hardware Failure

When a hardware failure occurs in the fabric, faults are raised in the system dashboard and are presented to the administrator. For cases where there is a component level failure with redundant components present in the system, syslog messages and SNMP traps are generated.

Examples of hardware events that generate syslog messages and SNMP traps include:

- Linecard failure on a spine switch
- Supervisor failure on a spine switch
- System controller failure on a spine switch
- Power supply or fan failures on a leaf or a spine switch

While Cisco Application Policy Infrastructure Controller (APIC) is a central point of management for the entire fabric, operations teams can leverage their existing NMS tools. Logging messages can be sent to syslog servers, such as Splunk, or SNMP messages can be sent to NMS systems, such as ZenOSS, to provide alerting. The leaf and spine switches in the ACI fabric also support traditional methods of detecting failures, such as SNMP polling at a set interval. If responses are not received from the switch in a certain timeframe, there is a possibility that the hardware has failed.

However, while the leaf and spine switches report SNMP and Syslog messages for component level failures, the APICs themselves do not have the ability to generate alerts using SNMP or syslog. For example a power supply failure on the APIC will not generate an SNMP or syslog message and must be monitored and remediated using the APIC dashboard.

Diagnose Equipment Failures

ACI fabric provides bootup, runtime, and on-demand diagnostics to assess hardware health for several leaf and spine sub-systems

1. Boot-up tests run when switch, card boots up
   - These are typically ONLY disruptive tests
2. Health (aka On-going) tests run periodically
   - Can only run non-disruptive tests
3. On-Demand Tests run on specific ports or cards for troubleshooting
   - There are no defaults
   - Can be disruptive

Hardware Diagnostics and Replacement

There may be times when you need to replace failed hardware. The Cisco ACI fabric employs a combination of key software and hardware features that are specifically designed to reduce the mean time between failures (MTBF) and the mean time to repair (MTTR). Regarding hardware, there are several hot-swappable components on both the leaf and spine switches in addition to a few components that are fixed on the chassis. If a data center ever experiences some sort of power surge or sees a component of their switches go bad, the hot-swappable components enable them to replace failed hardware quickly and non-disruptively.

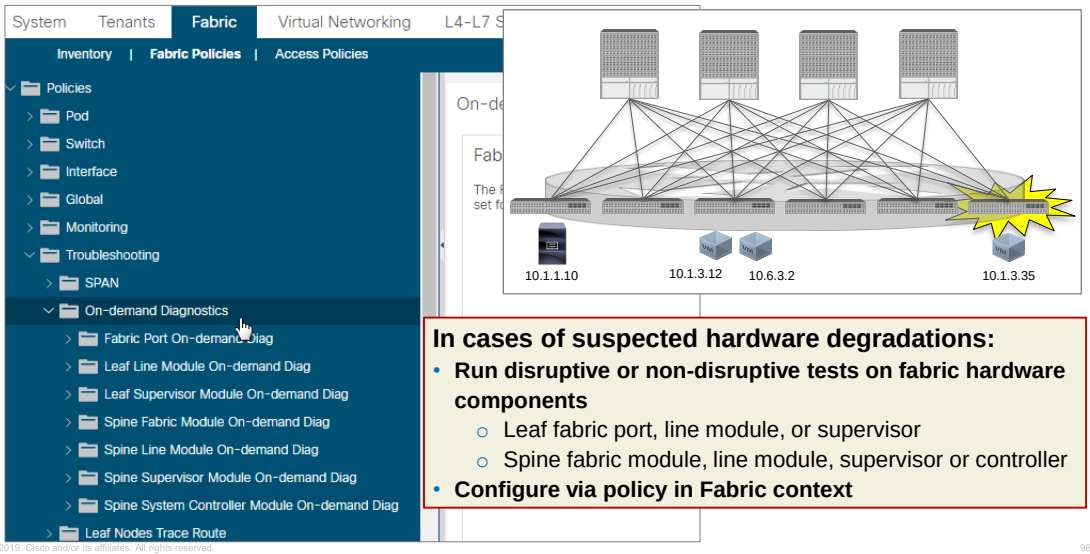
Note: The procedures for replacing hardware typically expect the new hardware to be the same as the hardware that you are replacing.

The ACI fabric provides bootup, runtime and on-demand diagnostics to help assess the hardware health of several sub-systems on each leaf and spine switch.

1. Boot-up tests run when switch, card boots up. These are typically ONLY disruptive tests. Comes with default set of tests that can be modified. Deployed via selectors.

2. Health (aka On-going) tests run periodically. Can only run non-disruptive tests. Comes with default set of tests that can be modified and are deployed via selectors
3. On-Demand Tests are to be run on specific ports or cards for troubleshooting, there are no defaults, and they can be disruptive.

On-Demand Diagnostics



In cases of suspected hardware degradations:

- **Run disruptive or non-disruptive tests on fabric hardware components**
  - Leaf fabric port, line module, or supervisor
  - Spine fabric module, line module, supervisor or controller
- **Configure via policy in Fabric context**

The On-demand Diagnostics panel contains a row of tabs. Each tab provides access to different types of on-demand diagnostic policies. Click on a tab to view a list of the chosen policy types in a summary table and to access an ACTIONS drop-down menu, which provides options to create or delete on-demand diagnostic policies.

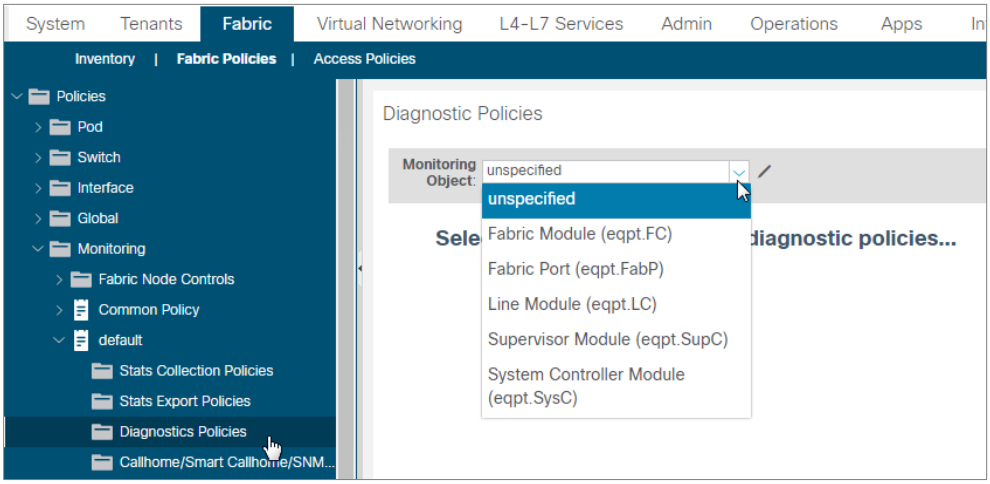
- The Create Leaf Line Module On-demand Diag Policy wizard enables you to specify a diagnostic on-demand test set for leaf fabric nodes to run on line cards (I/O cards).
- The Create Leaf Supervisor Module On-demand Diag Policy wizard enables you to specify an on-demand diagnostic test set for leaf fabric nodes to run on supervisor cards.
- The Create Fabric Module On-demand Diag Policy wizard enables you to specify a diagnostic on-demand test set to run on spine fabric cards.
- The Create Spine Line Module On-demand Diag Policy wizard enables you to specify a diagnostic on-demand test set to run on spine line cards (I/O cards).
- The Create Spine Supervisor Module On-demand Diag Policy wizard enables you to specify an on-demand diagnostic test set for leaf fabric nodes to run on spine supervisor cards.
- The Create System Controller Module On-demand Diag Policy wizard enables you to specify an on-demand diagnostic test set to run on spine system controller

cards.

- The fabric port on-demand diagnostic policy is a test set for fabric ports. The Fabric Port On-demand Diag tab displays the fabric port on-demand diagnostic policies as rows in a summary table and contains an ACTIONS drop-down menu with the options to create and delete fabric port on-demand diagnostic policies.

Configuring Diagnostic Policies

Diagnostics policy which acts as a container for associated fabric monitoring policies



Creates a policy which acts as a container for associated fabric monitoring policies. These can include policies related to Event/Fault severity, the Fault lifecycle, and other such monitoring policies.

1. On the menu bar, click FABRIC > Fabric Policies or FABRIC > Access Policies.
2. In the Navigation pane, right-click Monitoring Policies and select Create Monitoring Policy.
3. In the Create Monitoring Policy dialog box, type a Name for the policy and click Submit.

- Configure policies to specify what information is to be collected, such as statistics, faults, and events.
- Configure policies to specify how collected information is to be reported, such as by SNMP traps, Cisco Callhome reports, or file exports.

Discovery – Troubleshooting Challenge Labs

(DCACIO v4.0)

VM Ping Scenario

Launch all virtual machine consoles

- Invoke and leave running the following pings

App-server1	→	Web-server2	192.168.10.12
DB-server1	→	App-server2	192.168.11.12
Web-server1	→	L3ext OSPF	10.99.99.1
App-server2	→	Physical DB	192.168.11.200
Web-server2	→	App-server2	192.168.11.12

Each Tenant will launch the consoles for each 'virtual-machine endpoint' and invoke the listed ping scenario.

Break Lab Instructions – One Break at a Time

EXAMPLE – ODD Tenant Break

Break # > modify App BD Subnet to 192.169.11.1/24

Tenant***n*** | Networking | Bridge Domains | App | Subnet

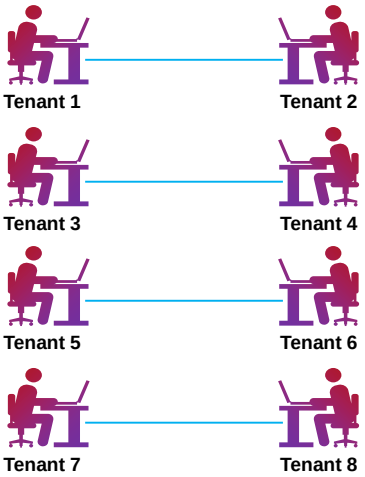
- Tenant1 > change Tenant2
- Tenant3 > change Tenant4
- Tenant5 > change Tenant6
- Tenant7 > change Tenant8

EXAMPLE – EVEN Tenant Break

Break # > delete T***n***-DB-port from Leaf103 profile

Fabric | Access Policies | Interfaces | Leaf Interfaces | Profiles > expand Leaf103-profile

- Tenant2 > delete T1-DB-port
- Tenant4 > delete T3-DB-port
- Tenant6 > delete T5-DB-port
- Tenant8 > delete T7-DB-port



The diagram illustrates the tenant parings. Each Odd tenant will break an assigned Even tenant (and reverse).

The break tasks are not symmetrical – i.e. the odd tenant's break task invoked on the even tenant is not the same as the even tenant's break task on the odd tenant.


CISCO